

MAKE | BUILD | HACK | CREATE

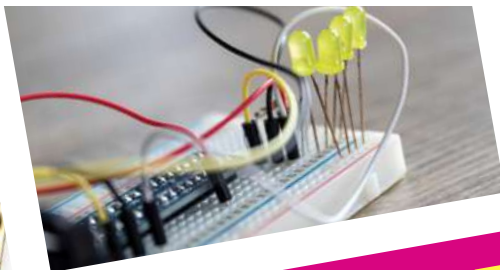
HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc

August 2021

Issue #45



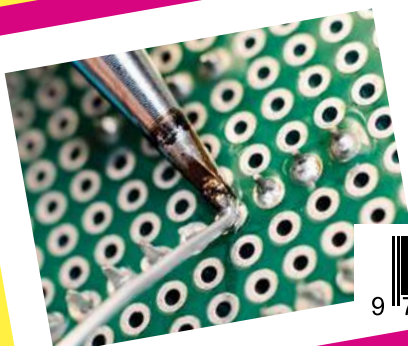
RASPBERRY PI BUILDS

Great physical computing projects you can make at home

+ DIY Gaming
Build your own
games machine

**+ Circuit
Sculptures**
Building electronic
artwork

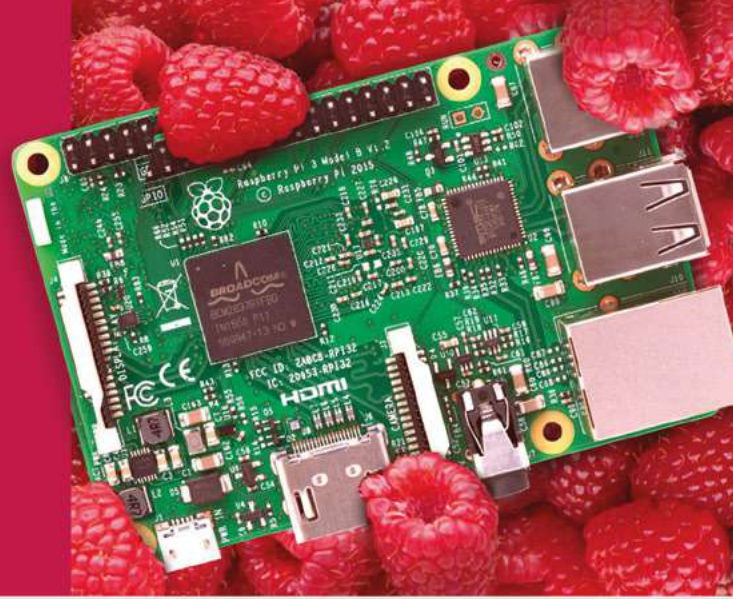
+ 3D Design
Build models
from sketches



Aug 2021
Issue #45 £6

WILLOW CREATIVE: 3D PRINTING IN COSPLAY

American Raspberry Pi Shop



- Displays
- Cases
- Project Kits
- Add-on Boards
- HATs
- Arcade
- Cameras
- Cables and Connectors
- Sensors
- Swag
- Power Options
- GPIO and Prototyping

Partner and official reseller for top Pi brands:



and many
others!

Price, service, design,
and logistics support for
VOLUME PROJECTS

USA



PiShop.us

Canada



BuyaPi.ca



Raspberry Pi

APPROVED RESELLER



Welcome to HackSpace magazine

When talking about flying machines, Antoine de Saint-Exupéry said that “perfection is attained not when there is nothing more to add, but when there is nothing more to remove”. If you apply this minimalist approach to computing, you get a Raspberry Pi. It's the bare essentials of computing, and from this you can build up whatever you want.

Because it's so stripped back, it's up to you how you build it up. What output device would you like? An HDMI screen? A DSI touchscreen? Something built up out of the GPIO pins? The choice is yours. Similarly with input – you can create whatever you like.

Raspberry Pi gives you freedom to create projects the way you want to, and we love seeing the creativity of the community in using this little powerhouse. In this issue, we're taking a look at some of our favourite builds. We'd love to hear what your favourite builds are. Let us know **@HackSpaceMag** on social media.

BEN EVERARD

Editor ben.everard@raspberrypi.com

Got a comment, question, or thought about HackSpace magazine?

get in touch at hsmag.cc/hello

GET IN TOUCH

hackspace@raspberrypi.com

[hackspacemag](https://facebook.com/hackspacemag)

[hackspacemag](https://twitter.com/hackspacemag)

ONLINE

hsmag.cc



EDITORIAL

Editor

Ben Everard

ben.everard@raspberrypi.com

Features Editor

Andrew Gregory

andrew.gregory@raspberrypi.com

Sub-Editors

David Higgs, Nicola King

DESIGN

Critical Media

criticalmedia.co.uk

Head of Design

Lee Allen

Designers

Sam Ribbits, Lucy Cowan,

Ty Logan

Photography

Brian O'Halloran

CONTRIBUTORS

Rosie Hattersley, Drew Fustini,

Sam Snyder, Jiri Praus,

Jo Hinchliffe, Marc de Vinck,

Alex Bate, Rob Miles

PUBLISHING

Publishing Director

Russell Barnes

russell@raspberrypi.com

Advertising

Charlie Milligan

charlotte.milligan@raspberrypi.com

DISTRIBUTION

Seymour Distribution Ltd

2 East Poultry Ave,

London EC1A 9PT

+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6, The Enterprise Centre,
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE

To subscribe

01293 312189

hsmag.cc/subscribe

Subscription queries

hackspace@subscriptionhelpline.co.uk



This magazine is printed on paper sourced from sustainable forests. The printer operates an environmental management system which has been assessed as conforming to ISO 14001.

HackSpace magazine is published by Raspberry Pi (Trading) Ltd., Maurice Wilkes Building, St John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2515-5148.

Contents



72

06

SPARK

- 06 Top Projects**
Creativity abounds in these homemade projects
- 18 Objet 3d'art**
How to never lose your SD card again: supersize it
- 20 Meet the Maker: Willow Creative**
Great cosplay starts with a hacked 3D printer
- 28 Letters**
On the false economy of free wood
- 30 Kickstarting**
Brilliant, unique wearable designs by Dr Kitty Yeung

33

LENS

- 34 Raspberry Pi builds**
Physical projects using the #1 tiny computer
- 46 How I Made: Ever-blooming Flower**
Brass, wood, electronics, 3D printing... and lots of love
- 52 Interview: Laura Kampf**
On tattoos, trolls, and joy of *The Simpsons*' intros
- 62 Improviser's Toolbox Milk cartons**
Face, flowers, Imperial Stormtroopers, and more
- 66 In the workshop Knife handle**
Working with wood that smells of sweet sweet Glenlivet

Cover Feature



BEST RASPBERRY PI BUILDS

Make wonderful things with the world's favourite computer

34



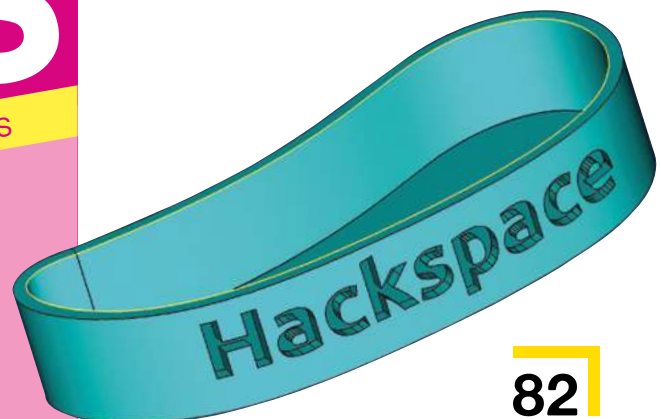
Tutorial

Build a game



88

As with Tetris, the simplest things are often the most complex



82



110

How I Made

Ever-blooming flower



46

The road to perfection is paved with broken prototypes

69

FORGE

70

SoM Raspberry Pi Pico

Control your Pico remotely

72

Tutorial Upgrade your CNC machine

Essential add-ons to make a good machine great

76

Tutorial Build an arcade machine

Add delicious vinyl decoration to the final build

82

Tutorial FreeCAD

Create fluid shapes with curves and extrusions

88

Tutorial Frustration box

Build a compelling game with just two buttons

96

Tutorial Nibble

Learn the nuts and bolts of video game design

Interview

Laura Kampf



52

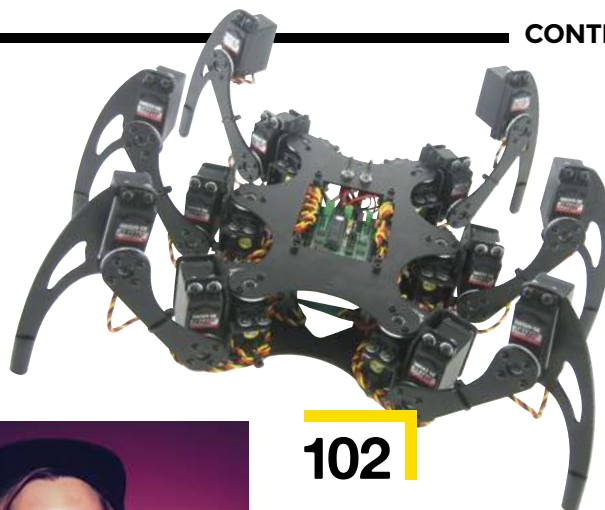
Why the workshop is Laura Kampf's happy place



06



30



102



76

101

FIELD TEST

102

Best of Breed

Walking robots to amuse/terrify

108

Review Pico VGA Demo Base

Add graphics and sound to your Raspberry Pi Pico

110

Review Nibble

Learn to program on this handheld gaming kit

112

Review EasyEDA

Design PCBs in your humble web browser

66



Some of the tools and techniques shown in HackSpace Magazine are dangerous unless used with skill, experience and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. HackSpace Magazine is intended for an adult audience and some projects may be dangerous for children. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in HackSpace Magazine. Laws and regulations covering many of the topics in HackSpace Magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in HackSpace Magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

Knitted keyboard

By Irmandy Wicaksono

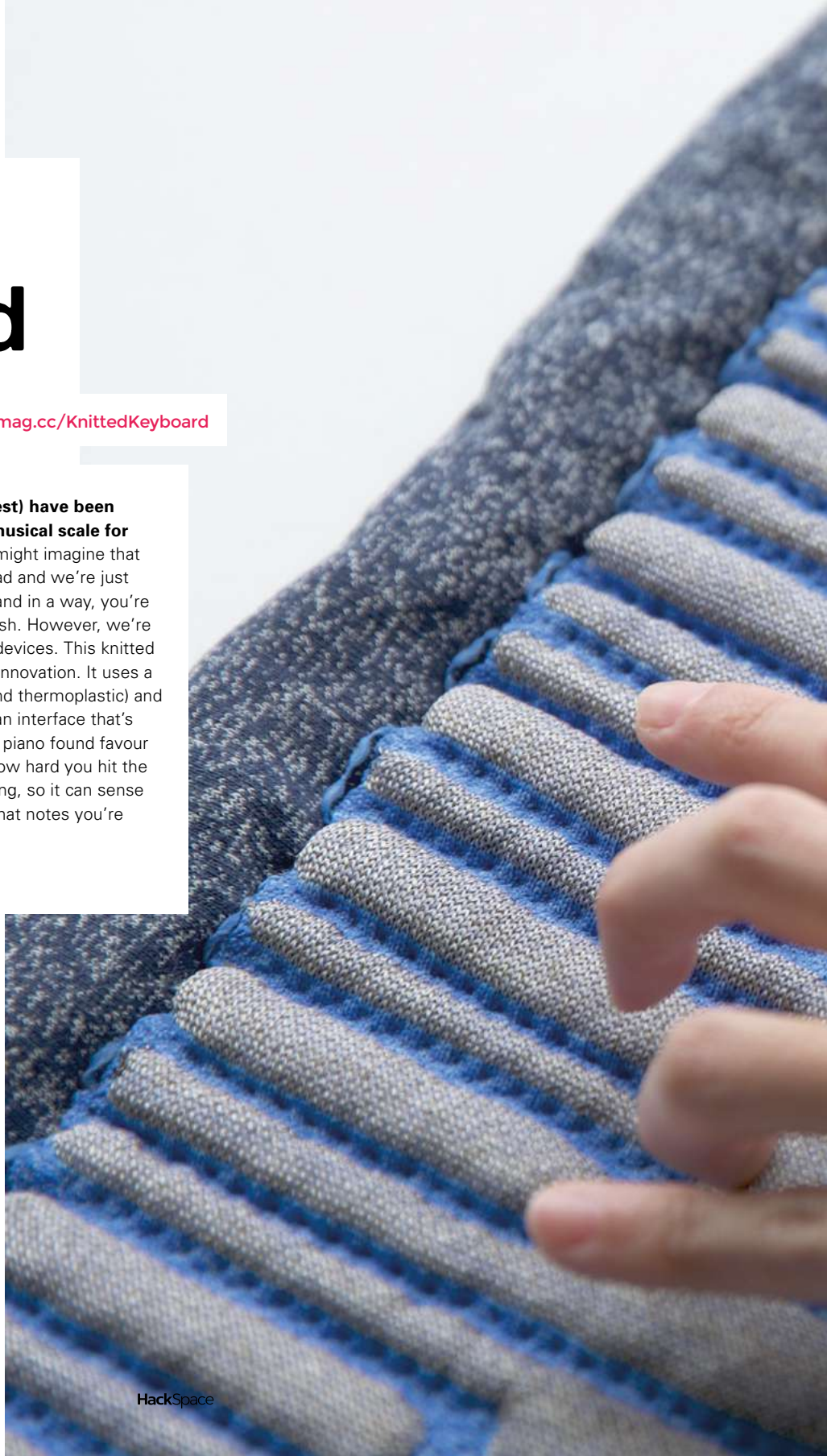
hsmag.cc/KnittedKeyboard

We (at least, we in the West) have been using the twelve-note musical scale for hundreds of years. You might imagine that all the ideas have been had and we're just recycling things by now, and in a way, you're right: chart music is rubbish. However, we're constantly innovating when it comes to input devices. This knitted keyboard from MIT is a great example of that innovation. It uses a combination of functional (conductive fibres and thermoplastic) and structural (polyester) fibres to give musicians an interface that's immediately tactile and familiar. And while the piano found favour over the harpsichord because it responds to how hard you hit the keys, this device also features proximity sensing, so it can sense when you hover above the keys and knows what notes you're about to play. □

Right □

You can even roll this keyboard up and wear it like a scarf

Image
Irmandy Wicaksono





Getula

By Ekaggrat Singh Kalsi

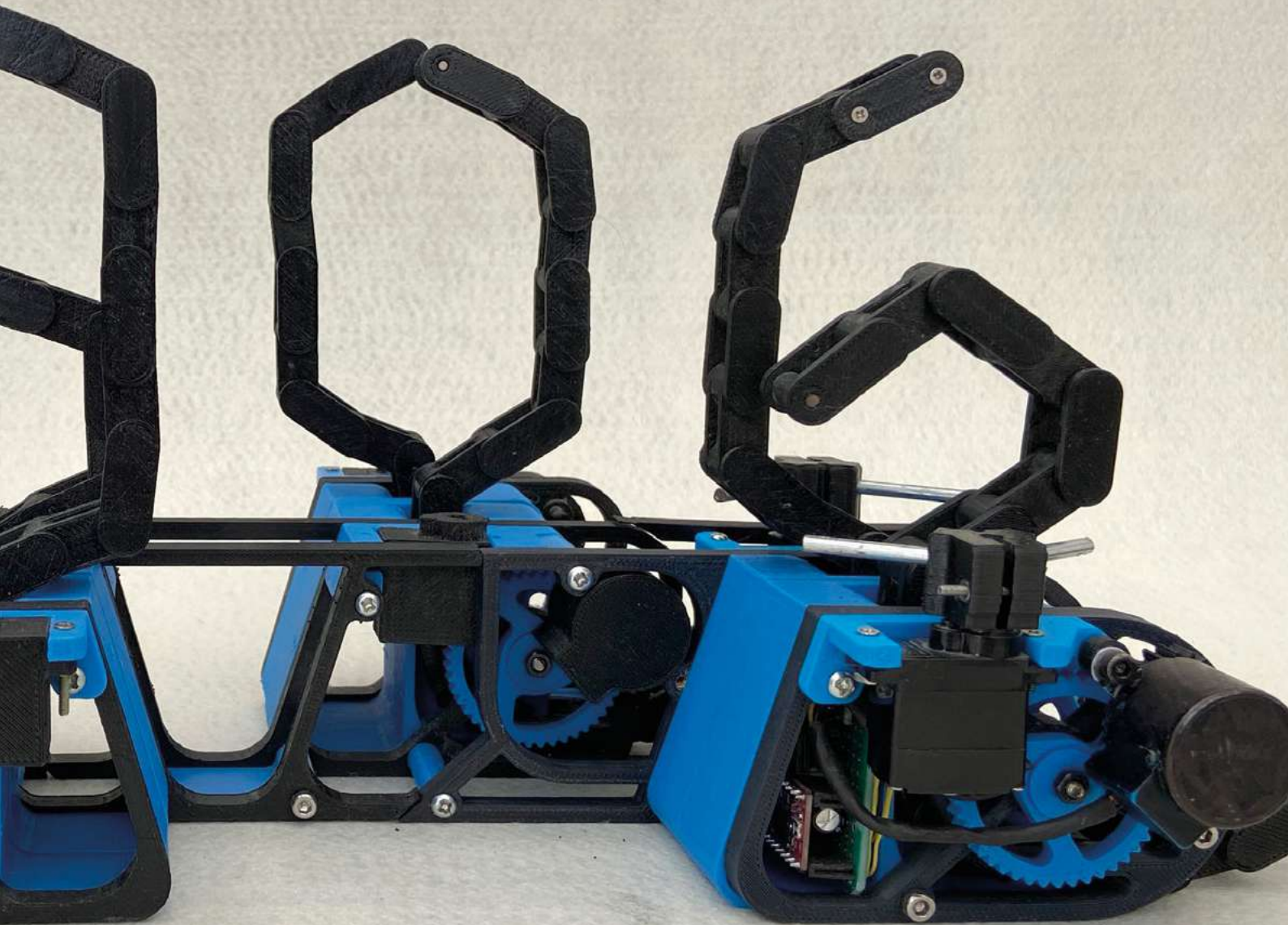
hsmag.cc/Getula

We've seen all sorts of displays over the years, many of which are iterations on old ideas. This clock, inspired by bicycle chain links, is unique. It's not, as far as we know, based on anyone else's work, and as such, it's a huge leap forward in humankind's ability to make cool things. Take a bow, Ekaggrat Singh Kalsi, Beijing-based architect and tinkerer and the creator of this prototype clock.

Each digit is its own self-contained module, and consists of a pair of stepper motors and two servos. The stepper motors push the chain up out of the base of the assembly, and the servos bend each link of the chain enough to form the shape of the number. It's a fascinating idea and, for such a mechanically inspired display, it's surprisingly organic to watch. ■

Right ■
The entire machine
is 3D-printed





LED pocket watch

By Spark Vulpes

hsmag.cc/LEDPocketWatch

Some while ago, Spark Vulpes acquired several broken watches with the intention of repairing them. Some repairs were successful; some weren't. But one in particular presented its own challenge: it had no internals. Most people would be reluctant to rip out old clockwork and put LEDs in, but as this watch was already just a case, it provided a perfect opportunity to do something different.

From the outside working in, the five rings of LEDs signify seconds, minutes, hours, and days (with Monday assigned the number 1); the final ring is used to display battery charge status. Working with such a small enclosure, and not wanting to modify it in any way, influenced many aspects of the design: most obviously the choice a 2.5 mm headphone jack to provide power, and the EFM32 Zero Gecko Cortex-M0-based chip designed for use in low-power applications. ▣

Right ▣

The watch case is sterling silver dating from the late 1800s / early 1900s, and now contains 157 LEDs





M&M sorter

By Jackofalltrades

hsmag.cc/M&Msorter

There's a wonderful legend that the 1980s band Van Halen included on their list of demands that they be served a bowl of M&Ms with all the brown ones removed. If they arrived at the venue they were booked to play and found brown M&Ms in the hospitality area, they were contractually allowed to cancel the show, at the expense of the promoter.

This sounds like spoilt brat behaviour, but the clause was inserted as a quick way to check that the promoter had read every line of the contract. Rather than check all the lighting, power, and sound requirements, they just had to look in a bowl of sweets to check that everything else was all right.

That approach wouldn't work today, as we have this visual sorting mechanism built by Jackofalltrades. It sorts a mixed load of M&Ms into six colours, using nothing more complicated than an Arduino Uno, a combination of a photoresistor and a white LED to detect the colour of the sweet, and a few other components to drive the mechanism. We'll admit, at first glance, we thought this was a machine learning project, but we're pleasantly surprised to find out it's so simple. ■

Right ■
Jack's machine can process 150 M&Ms in ten minutes





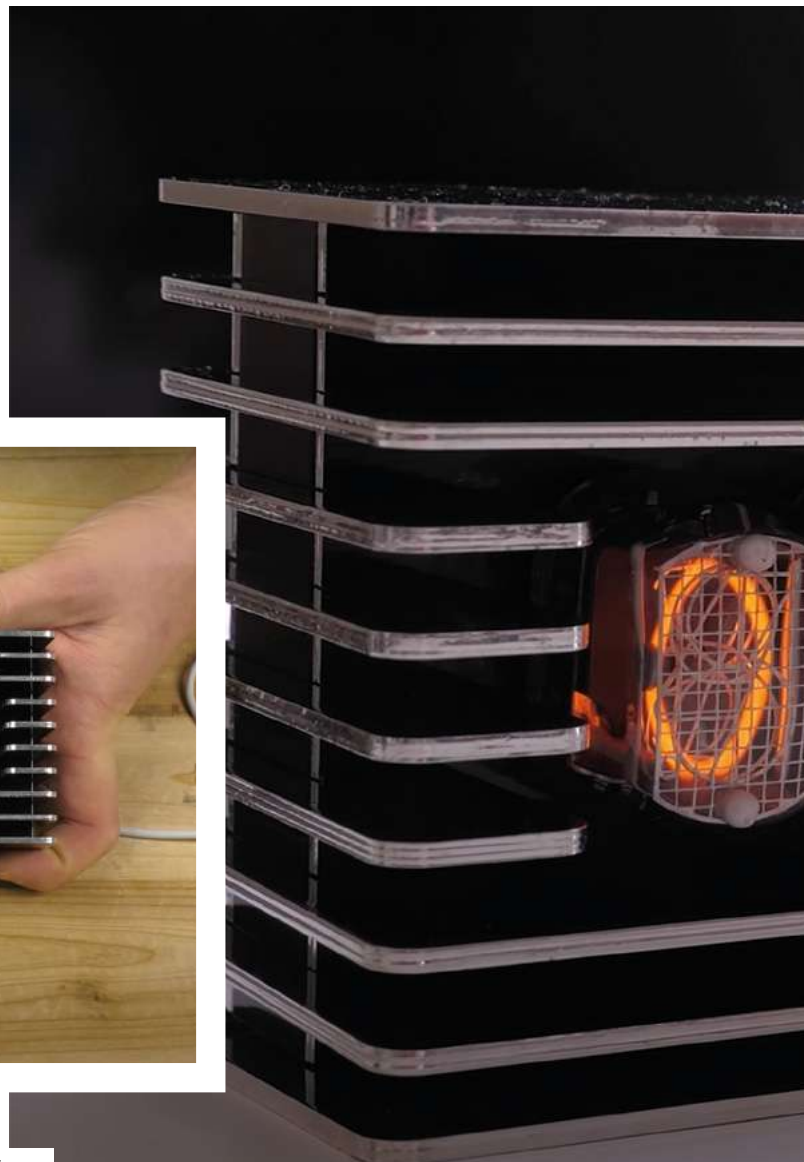
IN-12 Nixie clock

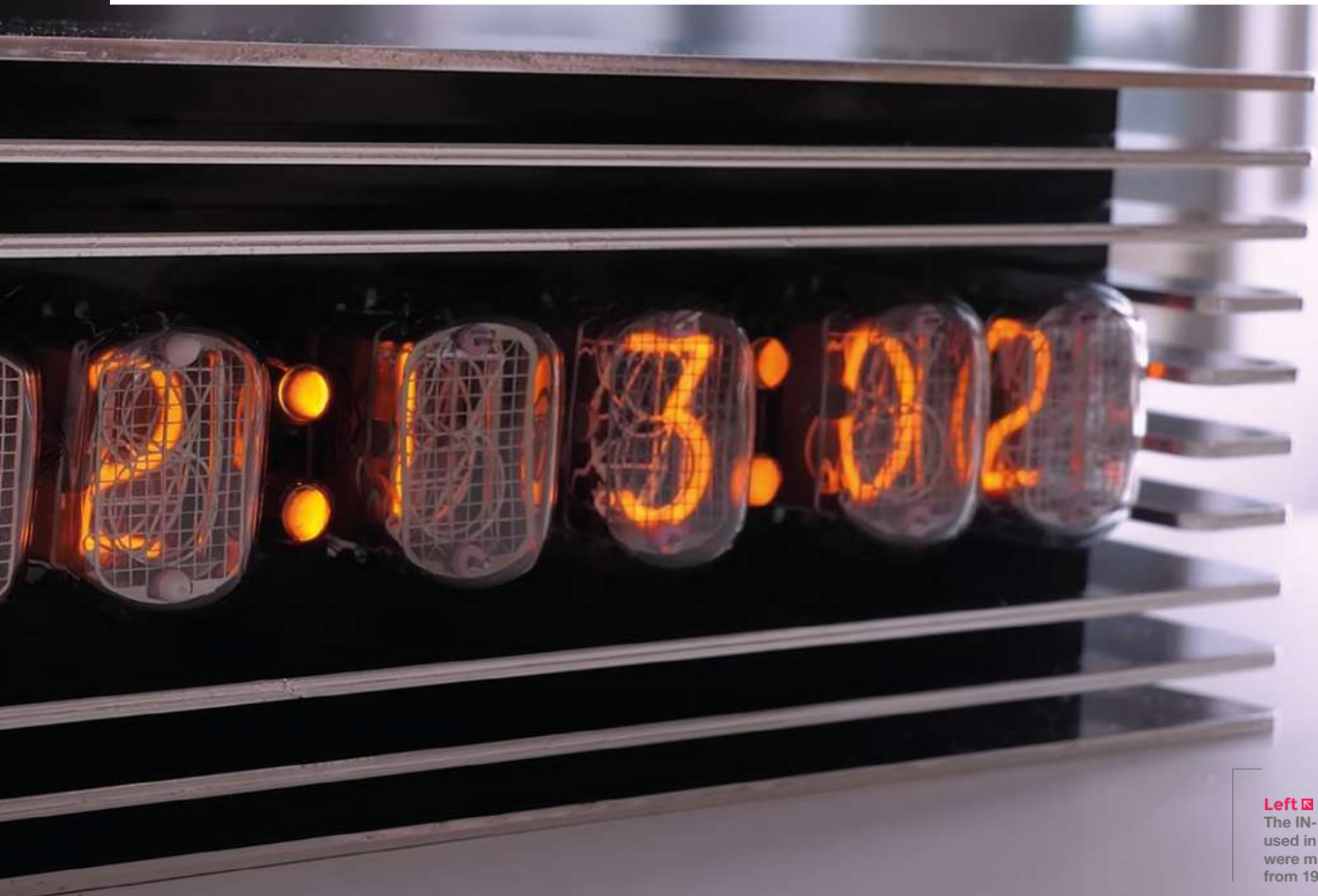
By Noyce Joyce


hsmag.cc/PCBNixieClock

We've featured the luxury of Dalibor Farny's brand-new Nixie tubes several times, as well as the work of Paul Parry, who uses Nixie tubes to breathe new life into beautiful old enclosures. What we have here is another take on the gloriously obsolete Nixie tube: a clock whose enclosure is also its PCB.

Featuring six original Nixie tubes and all the electronics you'll need to build it, this clock by Noyce Joyce comes as a kit. It's got all the bells and whistles you'd expect for something made in 2021, you can power it off a 5V USB supply, and it's open-source hardware. Go on, treat yourself. ▣





Left  The IN-12 Nixie tubes used in this clock were manufactured from 1976 to 1980

Tensegrity shelf

By Gammawave

hsmag.cc/TensegrityShelf

We've spent ages staring at tensegrity projects trying to work out how they stay upright. We work it out, then immediately forget, which means we get the pleasure of working it out all over again. This shelf is doubly baffling, as it was built to display the creator's other tensegrity projects.

The materials are all easily found in your local DIY store (the shelves themselves are made out of leftover bits of decking); for added cool points, Gammawave has used copper pipe with soldered joints for the solid bits. ▣

Right ▣

Luckily for us, Gammawave has written clear instructions for his latest tensegrity project





Objet 3d'art

3D-printed artwork to bring more beauty into your life

Fans of Jonathan Swift's 1726 masterpiece *Gulliver's Travels*, *Honey I Shrunk The Kids*, or Vic and Bob's giant frying pan gags will be well aware of the comedic potential of familiar objects scaled up (or down) to an absurd size.

Aarav Garg has used the weirdness of scale to great effect with this 3D-printed SD card, which also doubles as a case for a 512GB Seagate hard drive.

Aarav spent some time measuring a standard size SD card and inputting the measurements into Tinkercad before he realised that the measurements were already out there on the internet. After that, he scaled everything up by $\times 5$ (and customised the dimensions to make it deep enough to house the hard drive), printed out his own graphics, and turned the design into blue plastic. We're not sure why, but we like it. ▣

hsmag.cc/SDCard





Meet The Maker: Willow Creative

Putting the PLA in Cosplay



W

e'd never call ourselves expert makers here at HackSpace Towers: if we've managed to successfully bash two bits of wood together to make something that works, that's

a good day. If it also looks good, that's just a bonus. Merel Eisink, aka Willow Creative, doesn't take that view: she's a perfectionist. And it's not come out of a desire to master any particular technology – she's got good at making things because she's a fan of games, films, and TV. Rather than set out to be the best at 3D printing, the 3D printer is just another tool that she's used to express her love of culture, whether that's *Monster Hunter*, *World of Warcraft*, Marvel films, or just your common-or-garden werewolves.

“

I went pro as a maker two or three years ago now. It's my full-time job at the moment, mostly selling 3D-printed skulls

”

We spoke to Merel to find out what goes on under the skin of her costumes, why not having many tools is sometimes a blessing, and how anyone can get started in animatronics.

“I went pro as a maker two or three years ago now. It's my full-time job at the moment, mostly selling 3D-printed skulls, but also other 3D-printed props and other styles of masks that I've designed.

“I used to make everything with papier mâché and cardboard, which worked OK – I still use papier mâché for some parts, like horns. Then in 2017, I

was making a Demon Hunter costume, and I started learning 3D modelling and that kind of stuff.

“I thought, hey, what if I could use the 3D modelling design and get a 3D printer and make those designs [real], so I can take advantage of my schooling and the prop work and use a 3D printer to combine all that and make it a reality? That's how I started printing Demon Hunter horns for the costume at that time.

“The Demon Hunter horns got a lot of interest from other cosplayers, so I started printing for other people, started buying more printers, started a webshop, and then started selling more horn designs, masks, and skulls that I designed along the way. Sometimes people would email me saying, ‘I want this character; I have this design or character or idea and I'd like to know if you can 3D-print it for me.’ Usually it would be a different horn design or different type of mask design, and I'd 3D-print it for them and then put it in my shop. That's how most of the items in my shop got there – they're custom requests that I got, and now I sell them in my shop as well.

“I do still use [EVA] foam in some my projects, but I've been leaning more towards 3D printing for the last year or so, because it's a little bit more reliable in terms of symmetry of design. You can infinitely adjust your digital design before you touch any kind of material. That is really favourable for me as it's so much easier to iterate the design before you actually start destroying any of your materials.

“There are still a few designs that I make in foam because it's better for that use case – it's soft, it's lightweight, it's a little bit easier to bulk up and build bigger parts, because with 3D printing, it starts getting really heavy, and you still have to put in hours of post-processing to get it to look nice. ➔



Right 
Merel's latest costume, Fran,
is a character from the Final
Fantasy MMORPG games



"My next project will mostly be foam for that reason. And I just got a new laser cutter and I'll be using that on EVA foam.

"It's going to be really cool to be able to cut all those intricate patterns and still keep it digital, but do it in foam. It's going to be a lot lighter, more comfortable, and easier to assemble, without all the post-processing that 3D printing needs.

"When you look at other cosplayers who use 3D printing and crafting, most of the arguments against 3D printing in cosplay is that a lot of people use designs created by other people, so they don't design the costume themselves. A lot of the craft and the work, the thinking and the skill behind it, is taken out of the project if you use someone else's files, especially if those files are already optimised for printing and you can just load and go.

"At that point it's just a grind to do the post-processing: somebody else already did the work for you. I've done some judging for a cosplay contest before where they specifically took me on as a 3D printing expert, because I know how it needs to be designed, how it needs to be processed.

A SMOOTHER FINISH

"When you do 3D printing, especially on an FDM printer, you get all the 3D-printed lines, and those all need to be smoothed away. If you, as a judge, can see that something was 3D-printed, if you see the lines, and any other 3D printing artefacts, then you know that they haven't put in the effort to make it the desired finish that you're looking to achieve; especially if you're trying to do a metal finish, it really needs to be smooth.

"It takes so much time to do the sanding, because most of it has to be done by hand. I've tried a lot of multi-tools and bits and all that kind of stuff, but by hand is usually still the best. No matter how much filler you use, you still need to do the sanding. There's still a lot of elbow grease to be done if you want to avoid the 3D-printed look.

"Cosplay has got lots bigger in the last few years, especially with the international community stepping up with tutorials and videos and stuff that you can order online. A lot of people are getting into it and finding it via the online community.

"Before corona, we had about five to ten major events a year, so every month at least there was an event to go to, such as Comic-Con. You would always see cosplayers there, and almost everybody would be dressed up in some way. There would be tens of thousands of visitors for Comic-Con.

"The designs in games are absolutely perfect for cosplayers; they're so impressive, there are so many different elements that you can use to show off.



Fran uses a Raspberry Pi 4 to control her printer farm. Four USB ports = four printers



It makes for some really great costumes. The Monster Hunter costumes are fit to your character, so you don't have to make a complicated monster – it's just armour with extra extravagance. It's not like LARP [live action role-play] where you get a boring flat helmet. There's always a fun discussion between LARP and cosplay, because for cosplay you use a lot of different materials – you want to keep it as light as possible but also as crafty as possible, whereas in LARP they use the real stuff – metal and all the real techniques, all the traditional assembly methods they used in the Middle Ages. Whereas for cosplay, we innovate and make new patterns, and new ways to do things the easiest and the lightest way possible while still looking really awesome.

"If you look at a standard 3D printer, they're all about the same shape, right? They're all square. [In contrast], Delta printers have a round bed. The nice thing about the Delta printer is that they have a fixed bed, a fixed platform, so extending it is a bit more stable than if you were to try the same things with a bed that goes back and forth.

"You can also use a printer with a standard Ender 3, you can replace the lead screw, you can extend it to twice the height as well, there are some videos on YouTube about that. I've seen a couple of people do it with the CR10 from Creality, for example. So there are all the standard aluminium profiles; you can buy them online.

"You get a skyscraper effect if you're not careful. You have to be careful with what setting you use. It starts wobbling uncontrollably once it gets to the top. You have to print very slowly and make sure you don't extrude too much plastic and that the nozzle doesn't scrape on top of the print, otherwise it starts moving around.

"A lot of my prints have an additional pyramid shape at the bottom which gets cut away at the end, so it has a more stable base and doesn't move as much.

Above

The stilts in the werewolf costume are made from the same extruded aluminium that Merel used to extend her printer

"In 2019, I was making a new iteration of Demon Hunter, and they have really big light-up swords. At the time, I had three 3D printers, I think. There were a couple of ways that I could go about making the swords with the lights in them, but most of them were either going to be very complicated or very expensive or not that great to pick up or implement. The important thing was that it had to be a complete print, or it would not work.

"I researched Delta printers and ended up with an Anycubic Premier Delta printer; it's a slightly bigger model than the Anycubic Kossel printer, so I thought it would be more stable to use. I bought new profiles for it that are 1.5 metres long, I replaced all that, and now my Delta printer can print up to 1 metre in height. I made energy swords from Halo last year, and I made the light-up swords for my Demon Hunter costume.

"I also made a full-size model of myself on that printer to use as a custom mannequin. The legs were printed in one piece each on the printer. It was really convenient for my mannequin; I put in a larger nozzle so it would go faster, and it pushes quite a lot of plastic. The mannequin torso was a 24-hour print, and the legs were only eleven hours. They're hollow; I poured expanding foam into the cavity to fill up the space for a really low cost, so it worked really well.

"It's been very useful for my last project, and I'm definitely going to use it for my next projects as well.

"The stilts I incorporate into my cosplay are also made of the same aluminium profiles that I used to extend the 3D printer. I don't really have construction tools in my house – I just have a regular jig-saw, a multi-tool, and a drill, so that's all I have. The way 3D printers are constructed, they have sliding nuts and specific sizes of holes drilled for bolts. →

Below

When you're printing big things, you need a big printer





Above ♦
EVA foam is easy to work, light, cheap, and takes paint well (as you can see from the image below-right, which is the same costume at a more advanced stage)

“And I thought, ‘I could do that’. I’ve used wood before for stilts, which works, but wood gets bulky and heavy. The aluminium is great for that. With sliding nuts, for instance, I can customise the placement of my foot: I can move it further up or down for more comfort whenever I want to. I can add any extra parts for the stilts whenever I need. For example, I have some Velcro bolted to it right now so I can attach feet, skin, and all that kind of stuff for the costumes I use.

MOVING PARTS

“Everything I know about animatronics is self-taught from tutorials. I just started out with Arduino for the NeoPixels so I can download a couple of patterns, rainbows, etc., and install that in my props. And I also use DC motors from radio-controlled cars before.

“It was really nice, because they have a set rotation, a set rpm, and they keep rotating after you turn them on. After that, I started using servo motors because they are a little bit more controllable. For example, I used the servos in combination with my 3D-printed designs to create the links that open the eyes of an animatronic wolf’s head that I made, and shut them again. I watched a lot of linkage tutorials – common applications such as umbrellas and litter pickers opening and closing – to see how they work. If you pay attention to what is around you, you can see how



those things move and how you can transfer that to your own projects.

“It’s really cool to think about the real world and to see how much you can get away with. How do I place a motor there? How does it move? How much strength does it need to move it, and can I still fit in it?

“And then I use the Arduino Nanos to motorise them and do a lot of trouble-shooting with how much the servo needs to move in order to get the desired effect. Arduino is really convenient because you can trouble-shoot and make a new prototype if it doesn’t fit. For the servos, it’s pretty much the same as NeoPixels: you have a plus, a minus, and a data cable where the input goes. And then there are a couple of standard sketches for Arduino in the example menu... Basically, you can adjust all the parameters in that sketch and it already works, so it was really convenient.

“To move a servo is not that difficult: all you do is plug in the wires in the right place and then a piece of example code, you press Start, and it works.

“Before I started looking into it, I was really intimidated – it gets complicated when you start to make custom patterns for a certain costume, or you want to make a sensor or something. But if you have just standard movements, thankfully it isn’t hard to adjust a few parameters.

“Adafruit has a really excellent blog with all sorts of guides and projects. They also have free 3D printer files to make your own swords. You can make a lightsaber from their free tutorials, for example, and the cool part is they have a bill of materials, which includes an accelerometer so that when you swing it, it makes a noise. The tutorial shows you how to connect the accelerometer, how you accept the input of a swing, and then put it into your own prop – not a lightsaber, but maybe something that changes colour when you swing it.

“There’s so much content that you can adapt to your own projects, because they do so much on props and costume. You’ve just got to think about how you can implement their version of a lightsaber into your own prop or costume, get it set up, and adjust the parameters for your own projects.” □





Left ■
Next up for Merel:
Pepper Potts' armour
from *Avengers:
Endgame*



Build a Makerspace for Young People

Join our **free online training course on makerspace design** to get expert advice for setting up a makerspace in your school or community.

Sign up today: rpf.io/makerspace

BOOK OF MAKING

VOLUME 2



ONLY
£10

WHSmith
BARNES & NOBLE



THE BEST
PROJECTS FROM
**HACKSPACE
MAGAZINE**

THE ULTIMATE SKILLS,
TRICKS, AND MAKES

AVAILABLE NOW

hsmag.cc/store

FROM THE MAKERS OF **HackSpace** MAGAZINE

Letters

ATTENTION ALL MAKERS!

If you have something you'd like to get off your chest (or even throw a word of praise in our direction) let us know at hsmag.cc/hello

USEFUL ROBOTS

Before your interview with her in the last issue, I'd seen the work of Simone Giertz. I think it was a helmet with a toothbrush attached that went all around the user's face. Then there was the baby slapping machine – which was a motor with a rubber hand attached, which slapped a doll on its back. Essentially the same joke, and really, a bit tedious when you've seen it 10 or 20 times. And so on, for all the rest of the stuff she did. So thanks for the interview – not only did I get to revisit my opinion (one of the perks of old age is that you get lots of opportunities to admit you were wrong), but I also looked into the stuff she's been doing lately. The build where she's built a robotic uterus to simulate period cramps (and crush a drinks can) is a perfect example of how a skilled maker can use their powers for good rather than frittering them away on silly stuff. Long may it continue.

Rachel Wu

Toronto



Ben says: There's more to Simone's work than Shitty Robots, as she talked about when we interviewed her. But the thing I took away from that interview is that whatever you're making, there are always a consistent set of considerations. That's the lesson from Simone's earlier work; whatever you're doing, it's worth doing well. It's like Zen and the Art of Shitty Robots.



PALLETS

I read your roundup of projects built out of pallet wood with a wry smile. It's all well and good getting timber for cheap (or even free), but so many of those projects start with free timber and quickly move on to woodworking machines that cost hundreds of pounds and take up loads of garage space, which is something that I, as a renter in a small terraced house, just don't have. I pay more for the wood I use in my projects, but that means I've paid to have it squared and planed on all sides, which lets me spend less money on tools and working space. Not for the first time, I'm reminded of Terry Pratchett and Vimes' boots.

David Fuchs

Manchester

Ben says: Nothing you say here is wrong. If you want to get the best out of cheap wood, you really need access to the right tools. But you don't need to own them: you just need access to them. My local hackerspace has a planer and a jointer, so I can plunder cheap wood and do useful things with it without having to devote money and space to large tools in my house. It's such a good idea. I wonder if shared spaces with shared tools might catch on?



HEROIC FAILURE

Please extend my thanks to Fabian Steppat for his rollercoaster ride of a story about how he made a 3D-printed generator/windmill. It's so polished, so well designed, so perfect... but it doesn't work! On the one hand, what a letdown! But on the other hand, what an inspiration to see a project done by someone who is clearly far more knowledgeable than I am in every area that isn't perfect. People complain about Instagram only showing our successes and never our

failures, but that's not a new thing: it's natural to put your best foot forward and we've always done that, with or without social media. By showing off the huge amount of work that went into a project that ultimately didn't work, Fabian's done us all a huge favour and thrown down the gauntlet to the next maker who'll take on the challenge and eventually produce a 3D-printed generator.

David McLellan

Stockport

Ben says: Yes. 1000 times, yes. Any idiot can tell you that they did something and it didn't work; that's everyday life. But to go through, step by step, a project that consumed hours and hours of your life, and then discuss your failings... that's a rare gift. I've no doubt that we'll be hearing more from the WinDIY project. A 3D-printed machine for turning wind into electricity is exactly what we need, what with the sea on fire, Canada on fire, Australia enjoying a brief respite before it sets on fire again etc. Thank you and good luck, Fabian. No pressure!



CROWDFUNDING NOW

Art by Physicist

Combining tech with fashion

From \$49 | kickstarter.com | Delivery: August 2021

W

e've featured the work of Dr Kitty Yeung in the magazine before.

She's been innovating in the fashion technology space for years. You can see what she's been up to on her

Hackster page at hackster.io/kitty-yeung/projects.

She has been selling some of her fashion designs, but now she's running a crowdfunding campaign to increase the range of tech-enabled projects.

The new garments include a dress featuring lotus flower solar panels that can charge your phone, a coat with a built-in heater, and a dress with a WiFi-programmable LED matrix. You can see the full range on the Art by Physicist Kickstarter web page: hsmag.cc/artbyphysicist.


They're all stunning garments unlike any we've seen for sale previously, and pushing the boundaries of what's possible with tech and fashion. We often say with crowdfunding that you're not purchasing a product but supporting a creator, and this is a great example of this. Yes, you'll (hopefully) receive a fantastic piece of fashion, but you'll also help ensure that this nascent tech brand can continue to innovate and create inspiring clothing in new and interesting ways in the future.

The prices range from \$49 for a pair of brooches, to \$598 for a heated coat, putting this at the premium end of the pricing scale. However, for that money you get a unique fashion piece. Kitty also has a commitment to sustainable manufacturing, so you can be sure that your clothing is produced in as eco-friendly a way as possible. ▣



BUYER BEWARE

When backing a crowdfunding campaign, you are not purchasing a finished product but supporting a project working on something new. There is a very real chance that the product will never ship and you'll lose your money. It's a great way to support projects you like and get some cheap hardware in the process, but if you use it purely as a chance to snag cheap stuff, you may find that you get burned.

Left  You can get the LEDs arranged into any constellation



sinclair



Commodore

the

COMPUTERS

THAT MADE

BRITAIN



OUT
NOW

"The Computers That Made Britain
is one of the best things I've read
this year. It's an incredible story of
eccentrics and oddballs, geniuses and
madmen, and one that will have you
pinning for a future that could have been.
It's utterly astonishing!"

- **Stuart Turton**, bestselling author
and journalist

.....



Buy online: wfmag.cc/ctmb

Available on
amazon

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG
46

HOW I MADE: EVER-BLOOMING FLOWER

The quest for perfection takes many turns – but it's worth it in the end

PG
52

INTERVIEW: LAURA KAMPF

How on earth can one person be good at so many things? It's just not right.

PG
62

IMPROVISER'S TOOLBOX: MILK CARTONS

Cheap, malleable plastic that's ripe for experimentations

PG
66

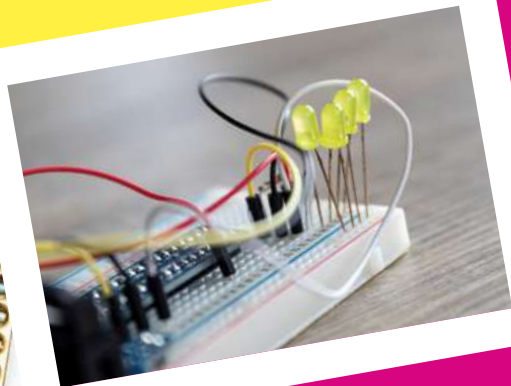
IN THE WORKSHOP: KNIFE HANDLING

Oak and steel: two-thirds of the things that built the British Empire

PG
34

BEST RASPBERRY PI BUILDS

Creative hardware projects to build at home



BEST RASPBERRY PI BUILDS

Great physical computing projects you can make at home

The world's most popular microcomputer turns up in the unlikeliest of places.
Rosie Hattersley introduces some favourite hardware-based builds



Raspberry Pi is at the heart of hundreds of projects that showcase the makers' imagination and practical design and building nous. Think of something you'd like to create and there's every chance an online search will turn up plenty of inspiration courtesy of like-minded tech hackers and makers. As you'll see from the following pages, Raspberry Pi can be used to entertain, to inform, to help enable explorations far into space, to bring the past into sharper focus, and to provide futuristic-seeming medical help.

The sheer wealth of successful projects we could have selected for this article is a great tribute to the flexibility of Raspberry Pi. Better yet, the sharing of details by makers and the community's amazing willingness to help others provides encouragement to take the plunge with your own ideas. Being able to tap into the community's experience, and read up on what did and didn't work along the way, ensures you can get a flying start with your own project. →



LED GAMES TABLE

Many people's first Raspberry Pi project involves retro gaming, often using RetroPie and either a 3D-printed or homemade retro case reminiscent of a classic console design. However, we particularly like this alternative take on eighties gaming featuring a grid of 300 LEDs, for which full build instructions are provided. Obviously, games include Tetris, Pong, and chess, but it can also be used with retro console games controllers or a keyboard. The LED Games Table supports two-player mode, making this striking piece of furniture ideal for an evening's gaming with friends.

hsmag.cc/LEDTable



Above ♦ Build a retro games table that doubles as a funky art installation

INSTA-CLOCK

Below ♦ Insta-Clock's themed photos representing numbers make for a whimsical take on displaying the time

hsmag.cc/InstaClock



There are lots of interesting time-telling projects involving Raspberry Pi, some of which play up the quirkiness factor. The Insta-Clock caught our eye for its artistic take on presenting the inexorable march from morn to night. Most of the images that are used to indicate each number are photos taken from around maker Riccardo's adopted home city of Copenhagen, with others set up especially for use with the Insta-Clock. These days, almost all of us are intimately aware of our immediate surroundings, so you may already have noticed numerical shapes in objects around you that you could snap on your phone and corral for a similar project of your own. The clock itself is simply two picture frames placed side by side, each containing a Raspberry Pi and a display. Friend Andreas created a Processing script (github.com/AndreasRef/InstaClock) that checks the time and refreshes the images indicating the hour and minutes.

6
PLUS
Jul 2016

Wozniak

Wozniak

STORYTELLING RADIO

8 Bits and a Byte's Dane and Nicole post all sorts of zany, kid-friendly projects on their Instructables page.

Their Storytelling Radio comes with its own Cold War-themed back story, while the Raspberry Pi 3B+-based device is a voice-activated take on a 1980s Choose Your Own Adventure story. Using a Google AIY Voice Kit, Google Cloud, and parts from an original Telefunken Bajazzo TS radio, it now listens for input as well as broadcasting stories. "The radio reads the story to you and when you need to make a decision, you simply say it out loud and the story continues. Raspberry Pi is the control centre of the project, passing the data back and forth to all the separate components, making everything work together," they explain. What's particularly lovely is that other makers have been inspired by this build to create their own for use in educational settings.



hsmag.cc/StoryTellingRadio

Left
Write and star
in your own
voice-activated
Choose
Your Own
Adventure story

RASPBERRY PI FILM CAPTURE

Digitising old video footage is a time-consuming labour of love that most people would rather not take on, but a significant group of enthusiasts have been steadily making their way through old analogue films in a bid to bring early video memories back to life. Head to the GitHub page (linked above) and you'll discover the endeavours of Joe Herman, who has dedicated himself and his Raspberry Pi to the process. Footage from 8mm and 16mm films are played on a modified movie camera and captured by Raspberry Pi frame by frame. They are then sent to a more powerful computer for post-processing and enhancements to the image quality and colour, along with artefact removal.

To get involved in the project and start restoring footage yourself, Joe recommends joining the Raspberry Pi Film Capture Google group hsmag.cc/FilmCaptureGroup. To see the results of his restoration work, check out his Vimeo page: vimeo.com/jphfilm.

hsmag.cc/RPiFilmCapture

Left
Family film footage
from the 1960s
and 1970s restored
with the help of
Raspberry Pi



Home Movies, Reel 2c, Part 2: Herman Lake



m090 - Ann Willmott, Christmas 1963



Home Movies, Reel 3b: Teens, cars.



m073: Christmas 1971 at Herman Lake

PIKON TELESCOPE



hsmag.cc/PiKon

Astrophotography and Raspberry Pi photography are both popular, especially since the launch of Raspberry Pi's 12.3-megapixel HQ Camera last year. The PiKon telescope melds these two interests, being a 3D-printed scope that uses a standard Raspberry Pi Camera Module with the lens removed in place of the eyepiece in a Newtonian reflecting telescope. Maker Mark Wrig explains that the PiKon was created for The University of Sheffield's Festival of the Mind, but proved so popular it was crowdfunded before being offered as open hardware – the STL files for 3D printing can be downloaded from PiKon's Instructable. There are sizable online Raspberry Pi photography communities that will offer plenty of help, advice, and admiration of your resulting shots once you get started.

Left The PiKon attaches a Camera Module to the telescope

GEIGER COUNTER



Below Measure radiation levels in steampunk style

hsmag.cc/SteampunkGeiger

Steampunk designs can provide a visual wow factor, and this gorgeous Geiger counter certainly intrigues.

Maker Chris Crocker-White decided to build it having discovered some cheaply available radiation boards at AliExpress that give off a satisfying clicking sound but “don’t really do an awful lot”. He decided to up the ante a bit by connecting the board to Raspberry Pi, feeding the data to InfluxDB, and using Grafana to visualise it. This in turn led him to design the case, complete with custom-machined brass parts and woodwork, topped off with LEDs and Nixie tubes he was able to source from eBay. In keeping with the steampunk retro feel, Chris used Dekatron tubes and a mechanical counter, and included a physical flip switch to turn off the audible monitoring of background radiation when it wasn’t wanted.

Various clocks, e.g. hsmag.cc/WordClock, can be seen on this site: hsmag.cc/Top10PiProjects. The Arduino version can be seen in the latest HackSpace.

TRACK THE ISS

If you're a fan of all things space-related, you may come over all crafty at the news that you can build your own International Space Station tracker using a Raspberry Pi Zero W and some heavy sheets of card. The brainchild of Alec Short (@ginandtronics), Project Arthur is named after the satellite dish at Goonhilly Earth Satellite Station in Cornwall, which carried broadcasts of Neil Armstrong's "giant step for mankind" in July 1969. The tracker was created to mark the 50th anniversary of the Apollo moon landings and has a red LED that flashes when the ISS is overhead.

STEP 1

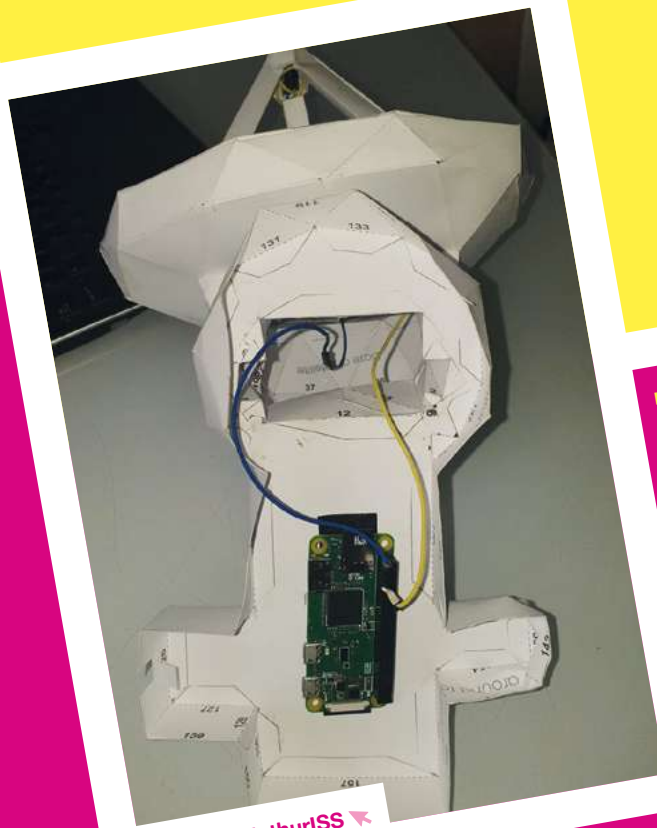
To build your own ISS tracking satellite dish, download Alec's GitHub code and instructions from apollo50.co.uk. Print the satellite dish design onto 160gsm paper; you'll need ten sheets of this thick paper stock. The papercrafting aspect is a little complex and takes a good couple of hours, so if you're taking on this build with a child, they may need help assembling Arthur.

STEP 2

The conditional web applet compares real-time tracking information from NASA and Open Notify about the space station's location with your whereabouts and pings your social media account to alert you when the ISS is overhead. This also triggers the LED. Register details of your location – your town or school, perhaps – plus your Twitter or Instagram account name at developer.twitter.com in order to interact with the NASA API and to receive these notifications.

STEP 3

Connect the LED and a resistor to the Raspberry Pi Zero W with jumper wires, attach a power cable, then place it inside the assembled satellite dish. You'll also need to thread the cables through



hsmag.cc/ArthurISS

the nose of the satellite antenna – another fiddly task. It's a good idea to cut out a gap for the USB power lead so the paper housing can sit flat.

STEP 4

With your Twitter account registered on the Developer page, create an access token and copy this access code to line 22 of the Python code you're using. Set up (or log in to) IFTTT (ifttt.com/discover), search for 'space' applets, and select 'ISS passes over a specific location' as the 'if' action trigger and pinpoint the location on the online map.

STEP 5

To create the 'that' action, choose 'Post a tweet' and add an 'ISSOverhead' hashtag to both the end of the applet and to line 6 of your Python code. Create a test tweet including this hashtag to check everything is working. →

Left Create an ISS tracker in a paper satellite dish

YURI 3 MARS ROVER

hsmag.cc/Yuri3



Airbus engineer John Chinner designed and built this incredible Mars rover to show to children while touring schools as part of his STEM ambassador role. The accuracy and attention to detail of this project can partly be explained by the fact John was able to sneak peeks at Airbus's prototype ExoMars rovers being tested in the firm's Mars Yard – a pretty cool perk of the job. He's also been honing his own robot-building skills since his teens. He was gifted the gold thermal blanket that gives Yuri 3 its distinct look by his Airbus colleagues. The fabric is usually used on satellites.

The gold thermal blanket ensures the rover can maintain a working temperature despite Mars' extreme temperature variations

The individually steerable wheels allow Yuri 3 to 'crab' sideways

This incredibly agile rover is able to spin on the spot and can be used in either tank steer mode or controlled remotely

Based on Beatty Robotics' Actobotics chassis, Yuri 3's six wheels are controlled by Hitec servo motors and a Raspberry Pi B+

WURLITZER JUKEBOX

hsmag.cc/Wurlitzer

Classic jukeboxes sell for thousands of pounds, and finding one to refurbish at modest cost isn't easy since they tend to get snapped up to retrofit and sell on. Maker

Marc Engrie happily got to work on this gorgeous Raspberry Pi rejuvenation of a Wurlitzer on behalf of his cousin, using Volumio streaming software – an app he was already using on his existing home Raspberry Pi setup. After stripping back and restoring the exterior and the speaker – which was all that could be salvaged from the inside – Marc added a 2GB Raspberry Pi 4 with HiFiBerry DAC+ Pro so music could be played from a USB stick. A new subwoofer system and LEDs to replace the original lights transformed the rather forlorn jukebox into a modern-day music-playing maestro.



Right ✦ An abandoned refurbishment project came back to life thanks to a sensitive restoration and the addition of Raspberry Pi 4 and HiFiBerry DAC

BUILD A RASPBERRY PI MUSIC STREAMER

hsmag.cc/PiMusicStreamer

Listening to music around the home on multiple speakers often means putting up with them being out of sync.

Raspberry Pi 4B and a JustBoom DAC HAT can help keep things in sync. You'll need a 7-inch touchscreen to control everything.

STEP 1

Mount the DAC HAT on the GPIO pins and screw in place, then connect the touchscreen to the Raspberry Pi. Update your Raspberry Pi to the latest version of Chromium.

STEP 2

Remove the case's wall-mounting screw holes and anything blocking the phono sockets. Line up the HAT with the top ventilation spacers, then place the touchscreen in the case.

STEP 3

Download the DAC HAT's drivers and install and configure Mopidy (instructions in link, above) as well as the Iris playback interface. Upload your music to Mopidy's Music folder, then edit Mopidy's configuration screen so it knows where to find music to play.

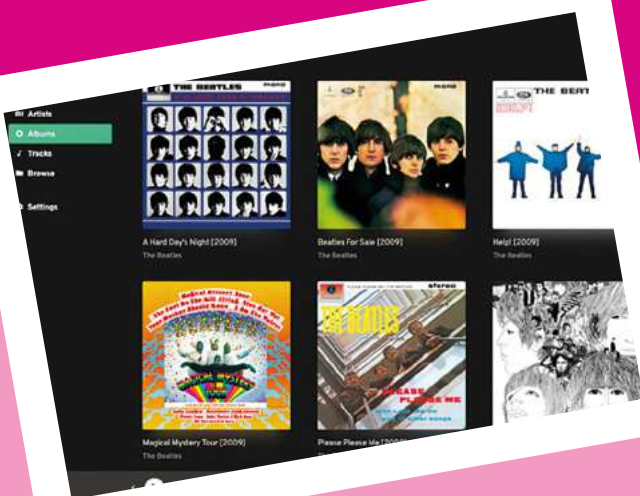
STEP 4

Restart your Raspberry Pi and launch Mopidy to check your music is showing. Run Chromium in 'kiosk' mode per the instructions given so Iris appears as the sole app and can be used to control playback. →

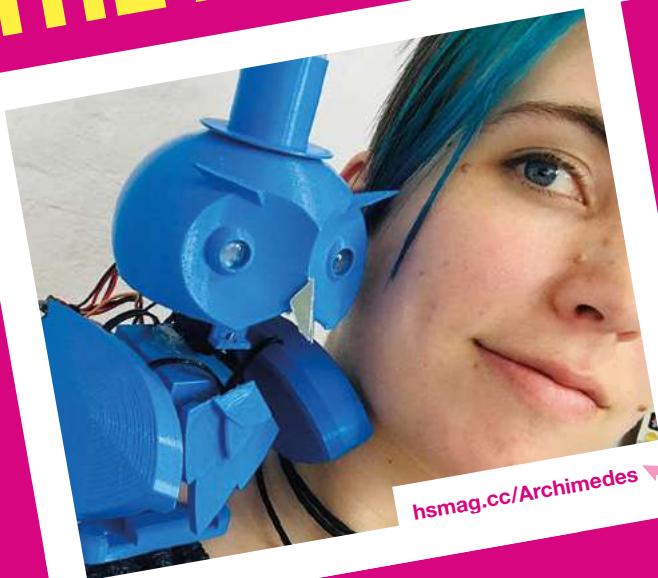


Above ♦ Stream perfectly synced music to multiple rooms with a JustBoom DAC HAT

Left ♦ Iris provides an excellent touchscreen music selector



ARCHIMEDES THE AI ROBOT



hsmag.cc/Archimedes

Back in 2018, Hackster's Alex Glow built Archimedes, an incredible robot companion using a combination of Raspberry Pi Zero W and Arduino with the Google AIY Vision Kit for its 'brain'. An updated model, Archie 2, using Raspberry Pi 3B, ESP32-powered Matrix Voice, and an SG90 micro-servo motor saw the personable owl familiar toughen up – Alex says the 3D-printed case is far more durable – as well as having better voice interaction options using Matrix HAL (for which installer packages are provided for Raspberry Pi and Python), plus Mycroft and Snips.ai voice assistant software.

Other refinements included incorporating compact discs into the owl's wings to provide an iridescent sheen. Slots in the case allowed Alex to feed through cable ties to attach Archie's wings, which she says now "provide a lively bounce to the wings, in tune with his active movements (as well as my own)."

Above ▣
Archimedes the familiar AI Robot has had a serious upgrade

PETOI BITTLE

Ready-to-play robot cats and dogs can be picked up from toy stores these days, but Petoï Bittle differs in that it's both a much more advanced model and it's been designed as a modular robot specifically so its owners can learn to build their own robots. Raspberry Pi controls the AI elements such as image recognition and tracking, while Arduino gives the quadruped's legs surprisingly realistic motion. Bittle can walk, crouch, and lurch forward in pursuit of the ball he's been tracking with his built-in camera eyes. He's also able to avoid obstacles and to get back on his feet if he has a tumble. Downloadable GitHub programs help him and his owner learn new tricks, and there's even a Bittle dog show. Ah!

hsmag.cc/Bittle



Above ▣
Bittle is a palm-sized robot pup that helps you learn Raspberry Pi robotics

ACTIVE MUSCLE STIMULATOR

Thomas Rask Thomsen describes himself as a 'cyborg human' due to the artificial leg he's upgraded with a Raspberry Pi 2 and uses to stimulate some of his leg muscles. Having

been born with a hereditary neurological disorder and with spasms and walking becoming difficult, Thomas was keen to use his interest in technology to find a way of keeping his muscles active and enabling him to get back to running. His Active Muscle Stimulator not only helps fulfil this ambition but is 'active' in the sense of actively learning from his body's movements, and using this information to decide when to activate his muscles.

Thomas has recently been sharing details of his latest version after creating a functional prototype last year. He hopes others will use them as a blueprint to build similar devices for themselves using his schematic and readily available items available online. "I will probably spend some

time collaborating with other creative people on adapting the device in various ways, such as enabling support for additional hardware of various kinds. But mostly, I hope to be busy running a lot," he blogged at the time. In April 2020, he reported having run his first 5km in years. By the end of April 2021, Thomas had built a new Active Muscle Stimulator based around Raspberry Pi Zero W, run 215km using the system, and was looking forward to reaching his 250km milestone.



blog.trask.dk

The Active Muscle Stimulator differs from some other such activation devices because it is able to learn how to translate body movements into muscle activation

Components for this initial prototype: Raspberry Pi 2B, Relay and Sense HATs, a battery pack, and TENS unit

The Relay HAT turns on and off the signals generated by an EMS (electrical muscle stimulation) unit

A Sense HAT monitors movement including rotation and acceleration, helping to determine when muscle stimulation signals should be triggered

MIDI FIGHTER CONTROLLER

Featured in HackSpace magazine issue 43, this MIDI Fighter tutorial showcases what a great addition Pico is to the Raspberry Pi line-up.

The 16-key arrangement makes use of Pico's generous complement of GPIO pins, while the arcade-style box is ideal for idly playing with sounds or for playing proper compositions. An Adafruit STEMMA expansion board and screen make up the rest of the hardware, which works with or without a digital audio workstation. □

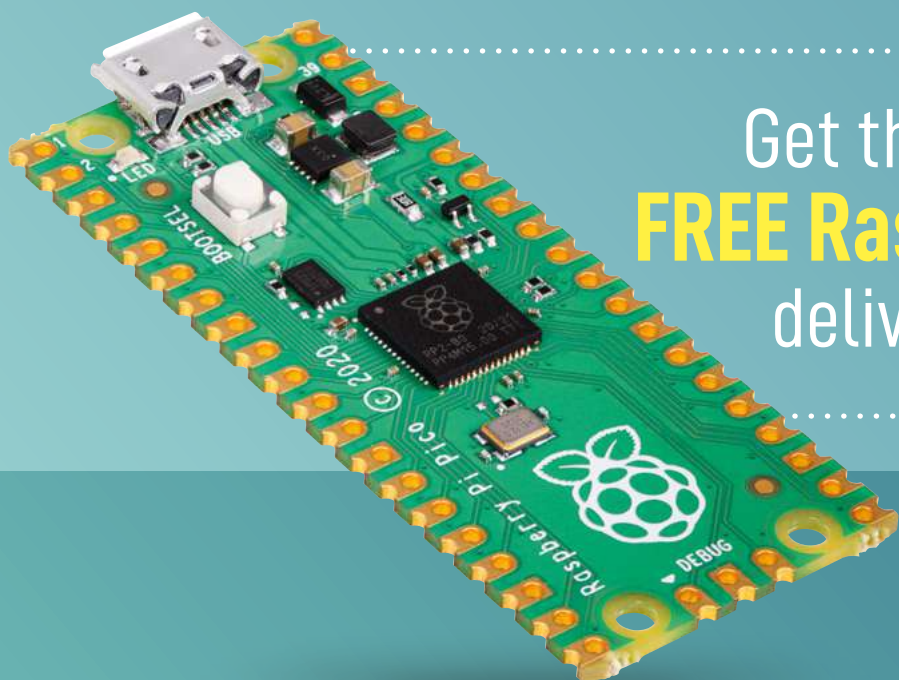
Right ■
Hammer those chunky arcade buttons to play music

hsmag.cc/MIDIFighterMusic



SUBSCRIBE TODAY

FOR JUST £10



Get three issues plus a
FREE Raspberry Pi Pico
delivered to your door

..... hsmag.cc/FreePico

UK offer only. Not in the UK?
Save money and get your
issue delivered straight to your
door at hsmag.cc/subscribe.
See page 66 for details.

Subscription will continue quarterly unless cancelled

SUBSCRIBE on app stores

From £2.29



Buy now: hsmag.cc/subscribe

Free Pico with print subscription only



MAKE | BUILD | HACK | CREATE

HackSpace

MAKE | BUILD | HACK | CREATE

TECHNOLOGY IN YOUR HANDS

hsmag.cc | August 2021 | Issue #45

SAVE 44%

RASPBERRY PI BUILDS

Great physical computing projects you can make at home

- + **DIY Gaming**
Build your own games machine
- + **Circuit Sculptures**
Building electronic artwork
- + **3D Design**
Build models from sketches

WILLOW CREATIVE: 3D PRINTING IN COSPLAY

MAKE | BUILD | HACK | CREATE

TECHNOLOGY IN YOUR HANDS

hsmag.cc | June 2021 | Issue #43

DEBUGGING
Make your microcontroller project work

INTERNET OF THINGS PROJECTS

SPACE KNOTS
Learn how NASA ties Mars rovers together

CYBERDECK ROBOTS

3D PRINTED WIND FARM
Generate your own electricity

CNC MITRE SAWS

PALLETS
Creative uses for budget wood

Use your Arduino Raspberry Pi Pico EVERYTHING

How I Made **FLOOWER**

From prototype to manufacturing –
a road paved with failures

By Jiří Praus

“Happy Valentine’s Day, honey. Here’s an ever-blooming flower I made for you.” That happened two years ago when I made my first

ever-blooming flower to bring mechanics into my electronic art. Today, I am running a small business handmaking and selling these to folks who love its uniqueness and minimalistic design, yet smart capabilities. Life is good. Or is it? There is a two-year gap between these events that was not filled with joy all the time – yet I do not regret it.

It all started when I posted a video with the blooming flower on the internet. Overnight it went viral. It had two million views in just a few days – even local news was talking about it. That was crazy. But nothing huge happened, as you would probably expect. I did not receive any contract or inquiry to start making those. I was naive. As fast as it started, it stopped.



If you want to achieve something, you have to do it on your own. The original ever-blooming flower took me two weeks to make. That was something nobody would like to pay for. I slowly started to redesign the project to be able to handmake it faster and make it more reliable. The most intricate part of the Floower is the mechanics of the petals, and the petals themselves. I designed them to close into a tulip-like shape with a slight overlap of petals – three inner and three outer petals. It looks pretty natural. I had to solder each from eight pieces of bendable brass wire. All of the petals have to be identical for the blossom to close nicely and evenly. To add even more complexity, I decided to put five SMD LEDs on each petal, adding another six wires that have to run down the stem to the flowerpot. This was a nightmare to make and maintain; however, it looks awesome. Nevertheless, this complexity had to go. It happened to me several times that the LED power wires just detached



Left ◆ CNC machines are fascinating to watch

classic Flooker is the trickiest one. I tried several manufacturers and types. Each spool, even from a single manufacturer, varies slightly in colour, and humidity plays a huge role in the print process as well. Great quality and stability can be much easier achieved with coloured materials, at least from my experience so far. Each petal takes 45 minutes to print, so experimenting is very time-consuming. Overall, I am still fine-tuning the process to this very day.

BRASS WORK

Moving down below the blossom, the stem and leaf are made from brass wire. And that is the real beauty and jewellery-like artwork. This is the same story as with the petals. Everything has to be perfect here. If there's a slight misalignment or the joint is not perpendicular, the whole blossom does not close nicely. This is where I developed templates for my soldering job. You would never guess the material I have chosen for them: plastic. More specifically, I am 3D-printing these templates from PLA. No, I am not crazy. Yes, plastic has a low melting point, yet for soldering brass it's important not to overheat the joint, otherwise the joint does not look good and the brass changes colour. PLA can withstand this intensive →

from the petal, and I had to resolder it. Something that no customer would accept.

3D PRINTING

So I decided to go with 3D-printed petals. Not only can they be made on a larger scale, but they also nicely diffuse the light coming from inside the blossom. The printed layers make them look interesting and tempting to touch. And most of all, I can make a perfect blossom since they are printed identically. From idea to finished petals, it took me several months.

The biggest issue was the print quality. right now I get tons of compliments on the petal's print quality, but that came at a cost. It took me hundreds of hours and tens of failed prints to fine-tune all the parameters. I started with PrusaSlicer, switched to Cura, and back to PrusaSlicer – just to later realise, I only need to play with the speed of print and retraction of filament when advancing layers to achieve a vase-like pattern.

You know, I had to fail to succeed. The lower parts of

the petal with hinges are printed with 100 percent infill. The upper part is just one perimeter, making it only 0.8mm thick. I was astonished by how flexible and resistant it is. You can easily bend it so that the tip meets the bottom, and it causes no harm at all.

Choosing the filament was another long story. The clear plastic I am using for the

Right ◆ 3D printing enables the petals to be made in any colour



Right ♦
All the parts ready for assembly

yet short heat without significant damage. That was a great discovery. From that moment on, I could print these sophisticated and complex templates for various parts of Floower brass work on a 3D printer and maintain perfect shape. Each template has a U-shaped groove to click the wire in, so it holds in the perfect position.

Inside the stem, there is a pushrod connected to a ring with six rods that moves with the petals. When the pushrod is dragged down by the motor, it closes the blossom and vice versa. Originally I started with plastic rods. Do you know what happens when you are using a hot soldering iron near them? The plastic rods just bend or extend. The rods were also very thin and when the petals are stressed they stretch slightly, introducing imbalance to the blossom. Unlike the soldering templates, the PLA was not a good material here. It also does not look nice at all. I tried to solder these from brass wire, but it was too complex to maintain perfect shape. In the end, I discovered laser cutting. Today, the Floower contains six stainless steel rods.

As soon as the weak point was fixed, another one appeared. And this one was a bigger failure since it was after tens of



In total, I used 33 brass wires in 14 different lengths to make a single Floower.

Quite impressive, huh?

Floowers were already in the hands of my first customers. The aforementioned pushrod is attached to the centre of the ring with a single T-joint. Don't get me wrong: I've tested the design of the mechanism. I've built a test bench where I run 200,000 open/close cycles in total, without any wear. I was pretty confident. However, this was

not how you are handling Floowers. You are not just opening and closing them: you are also touching and pulling the petals. Over time, the single joint gives up. And my ever-blooming flower ceases to bloom. I was quite lost here. I tried several wire-based structures to reinforce the joint, but it didn't look nice and wasn't easy to make. In the end, I figured out that a small sun-like plate with six beams and a hole in the middle would be the best solution. The centre pushrod would be soldered from both sides of the plate, and the ring would be soldered by six joints, not just two. I connected with a guy doing a brass etching, and the 0.3 mm thick brass rosette was born. Brass etching is a great technique. I can make whatever shapes are needed from thin brass sheets (0.1 mm to 0.5 mm thick).

In total, I used 33 brass wires in 14 different lengths to make a single Floower. Quite impressive, huh? Can you imagine hand-cutting these? Pretty laborious. I watched myself cutting the wires and then designed a machine to replace me – an automatic wire-cutter. It can unwind the spool of wire up

Left ♦
The tiny motor and gearing that makes the floower bloom





Left ♦
The wooden base gives it a natural touch

to 1 mm in diameter, straighten it, and cut it to precise length. I designed the whole machine. The frame is 3D-printed, reinforced by aluminium extrusions. Depending on the length of the wire, it can cut 1000 pieces in 20 minutes.

WOODWORK

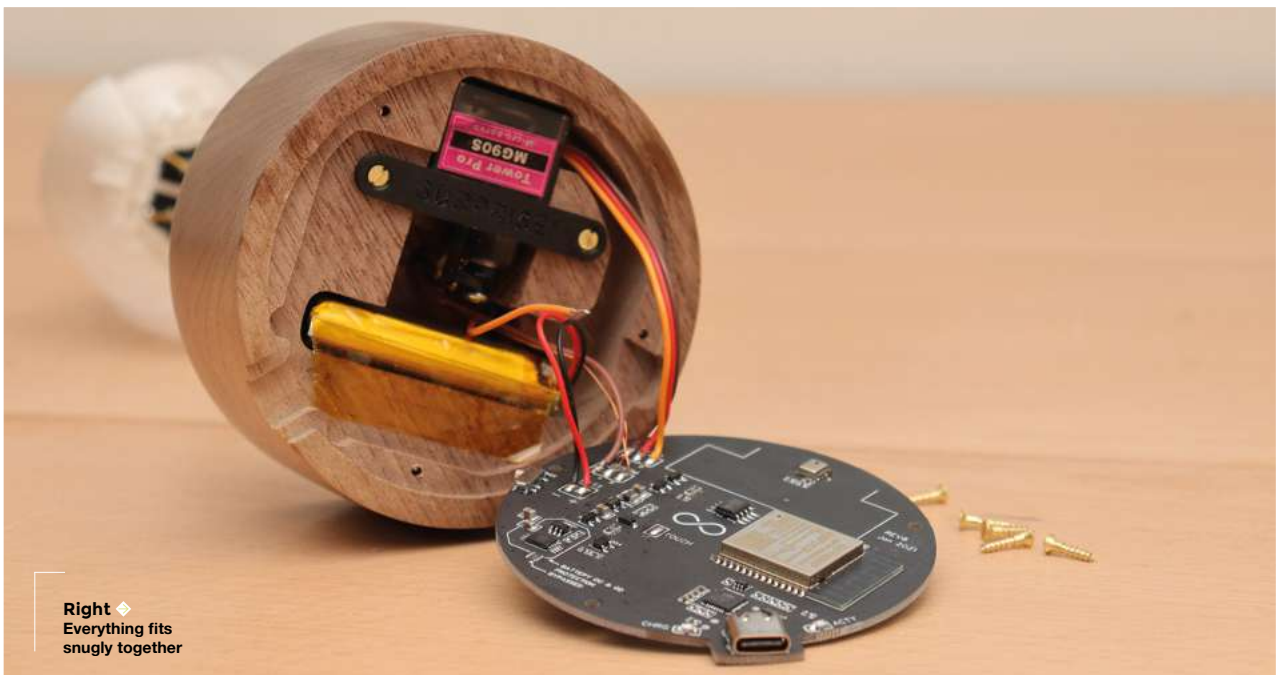
Wood is a beautiful material. I love the smell. I love the wood grain. I had to use

it for the flowerpot at the very bottom of the Floower. The flowerpot hides all the smart stuff the Floower possesses – motor, battery, and logic board. I made the original flowerpot from a piece of spruce wood, all by hand-carving out the centre with a router, making a big hole. All the components were mounted to random spots. Nowhere near what I would like to see from a professional product. This was a great excuse to get a

CNC milling machine. Or rather, to build it maker-style. I ordered a lot of components, designed and redesigned the frame and axes. After three months, the CNC machine was running. It would be easier to get a cheap one from China, but building machines makes me happy. Nevertheless, it allowed me to do unthinkable shapes. The flowerpot interiors are now carved in millimetre precision to perfectly fit the battery, motor, and logic board. No wood is wasted. I was also able to switch to oak, walnut, and other hardwoods. I am still an amateur here. I don't have a lathe. I am using my drill-press to sand the surface of flowerpots. Please don't judge me.


ELECTRONICS


I am a professional programmer but a self-taught electronics engineer. And the logic board was a huge challenge for me. I want my Floower to be smart – WiFi, Bluetooth, easy programming, various on-board sensors, battery-powered, and USB-C. →



Right ♦
Everything fits snugly together

FEATURE

Right 
It's a tight fit
getting everything
inside the base

Below 
The cog-like
part that links the
petals together



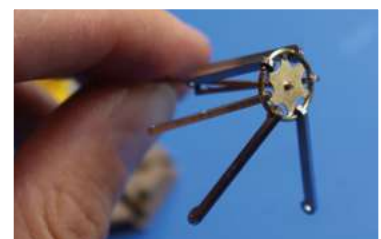
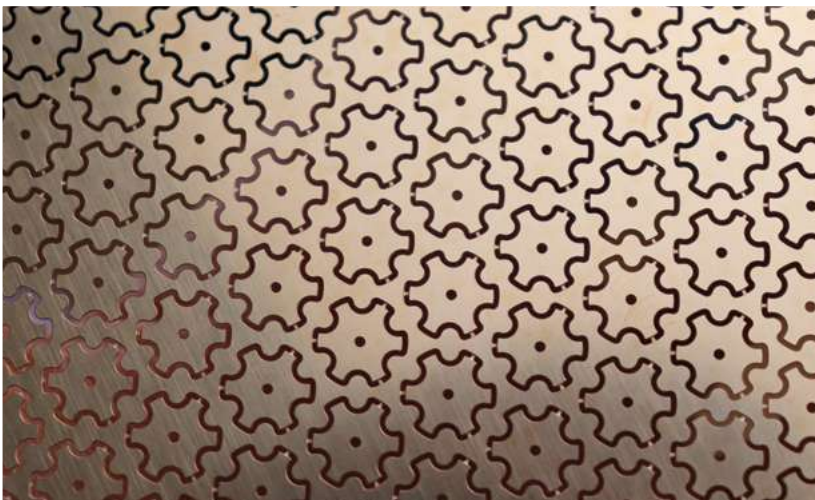
My first experience with a custom circuit board was with snowflake decoration. I've made several other boards since then, but nothing as complex as the Floower logic board. Anyhow, what I have learned is that PCBs are cheap these days. It's no more a luxury. There are tons of tools available for free where you can design your schematics and PCB and later send it to one of the hundreds of manufacturers.

So I started designing the logic board. I began with a development board and

modules. I wired everything together – ESP32 for MCU, USB to a serial module, TP4054 charging, battery protection module, boost-down converter to deliver 3.3V, and other passive components needed. When the setup was tested, I drew a schematic, designed a PCB, and sent it over to the manufacturer – hoping that I had not made any mistakes in the design. The opposite was true. This would probably make its own article: 'How I wasted money on failed designs'. The boards worked, but

there were always some small mistakes or improvements I discovered when they arrived. To cut a long story short, it took me six iterations to finally have something useful for the first Floowers:

1. Adding battery over-discharge protection
2. Removing boost-up converter for servo
3. Switching from CH340 to CP2104 USB to serial converter due to unexpected behaviours
4. Introducing MOSFETs to switch power off for LEDs and servo to save power during deep sleep because they consume power even though they are not doing anything





Left
The finished Floower. Just touch to light

It's great to pursue your dreams. Make.
Innovate. Just be patient – it takes a long
time to deliver something of great quality

5. Switching CP2104 to self-powered mode so that it's consuming power only when the USB is connected, not all the time
6. Following up on the specification for touch sensors routing on the PCB to decrease the noise of the capacitive touch sensor

SERVO MOTOR

I chose a servo motor to move the blossom. I designed a crank from a short wire that transfers rotary movement to linear. I had to experiment here a lot. I bought myself 20 different tiny servo motors from different manufacturers and, one by one, tested them out. I faced a variety of problems here. All the servos are rated for 4.5–6V, but since ESP32 is

running on 3.3V and the whole Floower is powered by a LiPo battery running normally something like 3.9V, this was a problem. I tried to introduce a boost-up converter to have 5V for servo, but it was too complex, and power consumption was too great. So I tested servos running on 3.3V; surprisingly, most of them were running. But some of them were not strong enough to close the blossom, some of them were shaking during movement, and some of them were too noisy. In the end, only one servo was strong and silent enough to meet all my requirements.

After 350 Floowers, I am abandoning the servo motor in favour of a stepper motor. Again it took me half a year to find the right one – one that would fit inside the flowerpot, and would be strong, smooth, and stocked enough. I've now

finished redesigning the electronics, and I'm introducing some environment sensors such as temperature, humidity, CO₂, and light intensity. The Floower will be fun! You know, never stop innovating.

CONCLUSION

To sum up, this small artwork of an ever-blooming flower looks pretty simple and minimalistic at first. Yet it combines seven manufacturing techniques I had to master: wood milling, 3D printing, soldering, etching, laser cutting, electronics design, and programming. Everything apart from the PCB is made in-house. There are lots of people finding Floower useless – luckily, there are a lot of you who understand what it really represents. The Floower is not only an exciting artificial flower, but also a model of innovation and makership. It's a combination of craftsmanship and technology.

It was and is still an exciting journey. It's great to pursue your dreams. Make. Innovate. Just be patient – it takes a long time to deliver something of great quality. If you have any further questions, feel free to contact me at jiri@floower.io. □

HackSpace magazine meets...

Laura Kampf

Woodwork, metalwork, and homemade tattoos

Laura Kampf, the Köln-based wood and metalworker with a mild tiny house and Leatherman obsession sat down (virtually) with Alex Bate to talk about prison tattoo

machines, avoiding your nightmares, and why aggressive hip-hop and horror movies inspire her weekly project builds.

Smudo the workshop dog was also there, which seems to be becoming a recurring and pleasant feature of these interviews. →

Right ↗

In five years, Laura has uploaded over 200 videos to YouTube



HS Your videos feel very unique in how they're produced. It feels as though we're in the workshop watching you get on with your day. That you'd be doing this regardless of whether the camera was there or not.

LK Yeah, that's absolutely it. I mean, I document it for YouTube because I'm aware that this is the only place for me. And the documentation, that's the work part, like setting up the camera, thinking about the story. But the physical work of building something, that's a form of meditation. That's just my happy place. And I know I have to document my work because I have to do something to make a living, right? I can't just play. So YouTube is my work. But making is just, it's just what I do, and I feel more and more that this is the only place for me.

And this is probably how musicians feel when they are performing on stage. You know, this – being in my shop, I feel so comfortable. And I feel so good. I don't have that anywhere else.

HS I remember seeing that you went to design school. Is that where your journey as a maker started or does creativity run in your family? I know your brother is creative ([instagram.com/zooburger](https://www.instagram.com/zooburger)), but what about your parents?

LK My brother is super-creative, but my parents, not so much. My grandfather was an engineer. So I think it kind of skipped a generation.

In design school, there was a project where we had to build something out of everyday objects. And it was for us, the designers, to get away from the computers and just do something with our hands. I built a tattoo machine, like a prison-style tattoo machine. And I was hooked. I remember coming home and I was so moved by the whole thing. Even though the machine looks terrible, everything fell into place.

Because all my life, people were telling me you need to find this one thing that

you're really good at and then just keep doing that. I think it's also a German thing, you know, like, be perfect at one thing, and then you'll be the best in your field. And I could never focus on one thing and building this tattoo machine; there were so many different things coming together.

I had all this interest in so many different fields and I could use them for the project – I enjoyed drawing fonts and learned how to do old-school tattoo lettering, and I could do a little bit of electronics to hook up a switch. All these things, I thought it was super-interesting. It was the first time I could just use little bits of everything I knew to make something that was really cool, and I was hooked after that.

HS Have you tattooed yourself with the tattoo machine?

LK I wanted to and then, thank God, because I was really young, it's very likely

I built a tattoo machine, like a prison-style tattoo machine. And I was hooked

that I would have done that, a tattoo artist came by and I showed him the machine, and he was like, "Don't do it. It's running way too fast. You will make mincemeat out of your skin". But I bought pigs legs and pig's ears and tattooed them. I couldn't eat pig for probably two years after that. It was so warm, and tattooing the piece for a couple hours, the fat was running out of it – it was disgusting.

HS It's interesting that if you look at the stuff that you're making now, the anchor point that started all of this is a prison tattoo machine.


LK Looking back, I remember the little things that I made; when I showed them to people, they just didn't show the same excitement for them as I did. And it was such a disappointment until I realised that no, the stuff I was making was really bad. That's why no one was excited, because I didn't know what I was doing. Once I got better and better, and especially with YouTube and talking to the community – well, I'm preaching to the choir here; everyone knows making is fantastic, and we have a very focused, niche community – and they get it.

HS Do you feel like a bit of a sense of responsibility being a woman in this community, being queer in this community? Two of the things that are a minority in this field. Do you feel that affects your work at all?

LK I didn't to begin with, I have to say. In the beginning, I felt more that it wasn't about me, it was about the things that I make, and my sexuality and my gender don't play a role in this. I don't think about my sexuality all day long; I don't think about the fact I'm a girl all day long, so why would it be in my videos? But I have to say that I changed my mind about these things. Because visibility is really important.

I had this really weird experience at the 10 Maker event a few years ago. I was wearing this T-shirt I got for free on one Christopher Street Day, it says 'Gay Okay'. I love that shirt; it's a really nice fit. I went to get some groceries with Brett from Skull and Spade and Hassan from HABU, and there was this girl, maybe eleven or twelve years old, and she saw me wearing that shirt, hanging out with regular dudes, doing regular stuff in a regular supermarket, and her jaw dropped. We were in the countryside, you don't see things like rainbow flags there. And I could tell she's maybe gay too, and it was so good for her to see that. There's nothing different about you – you can still hang out with guys, you can still, →



Left  Laura works mostly with wood and metal in her weekly videos

you know, go shopping and all these things. That's when I realised, institutions like Christopher Street Day are so important, but it's also important to just have it integrated into regular stuff, not just special occasions. Today's International Women's Day? Well, we need to celebrate girls every day; every day you need to celebrate these things.

So, I kind of made it a habit to have rainbow flags in my videos. Not every video, and never super-obvious, but in the background, when I talk to the camera sometimes. I do wear my Gay Okay shirt every once in a while. I don't want to make it a point because people like to put you in drawers. And, once you're the queer maker, you're the queer maker, and that's all people want to talk about. And I don't want that because I still think, at the end of the day, it's about the things I build and not about me and my sexuality and gender. But, yeah, to just sprinkle it in every once in a while, I think it's very important.

I don't get much negativity about this. I was surprised, pleasantly so, obviously, but yeah, a couple of days ago, I wore my Gay Okay t-shirt in my Instagram Stories, and people applauded me for it, and that's really interesting. I would never have thought that.

HS Do you get much trolling at all? Or are you spared from it?

LK I think, at the beginning of my YouTube career, I was growing really fast and really, like, exponentially. And I had a couple of videos that went viral, like the beer bike, that went outside of the community. For those viral videos, you get negativity. They don't know who you are, they don't know the context, they don't know what I'm doing. That's why I hate having viral videos. It brings in the worst. I like to be in this little lake, surrounded by my followers.

HS A few people have said that, actually. That it's the worst. It's the thing

everybody aims for and then, when you get there, you wish you weren't.

LK Yeah, they take you out of context. Those people, they see one of my videos, they don't know that I'm building something. And that's another interesting thing that your community learns about you. They know I build something every week for the past six years. Every week, it can't be the Holy Grail every freaking week. Sometimes it's bad, but it's stuff that I did that week – it's documentation.

When I was a kid, I remember my mind was blown that *The Simpsons* had a different intro every episode. Something different happens every time. I couldn't

For those viral videos, you get negativity. They don't know who you are, they don't know the context

believe that, and how much work went into it. I think it primed me for being a weekly creator.

HS It's impressive. There aren't a lot of makers releasing weekly videos, and many that do are releasing build videos in weekly parts. And you just come along and go ta-da!

LK Haha, but not every video is a good idea. Some of them are really bad ideas. But that's my privilege, you know, that I can still do that. Because I have to, otherwise there wouldn't be a video, and I love that because the pressure helps me keep going. And the process is the same. It doesn't matter if you're building a tiny house or a scratch post for a cat. The process is me, being in the shop, listening to podcasts, listening to music, enjoying my tools, playing with the material – it's all the same, it doesn't really matter.

So, the public bench stuff that I've been doing lately, I get so many questions like, "Oh no, how could you leave the bench" and, like, I don't give a damn about the bench. It's not the bench, it's the process that I enjoy. I could literally throw everything that I build away – I could throw it in the trash right away. I wouldn't mind. I'm so focused on the process.

HS I was going to ask you about the bench, because it was recently vandalised and so you made another one. Most people would probably just not, would just raise their hands in defeat and leave it. But you just made it again.

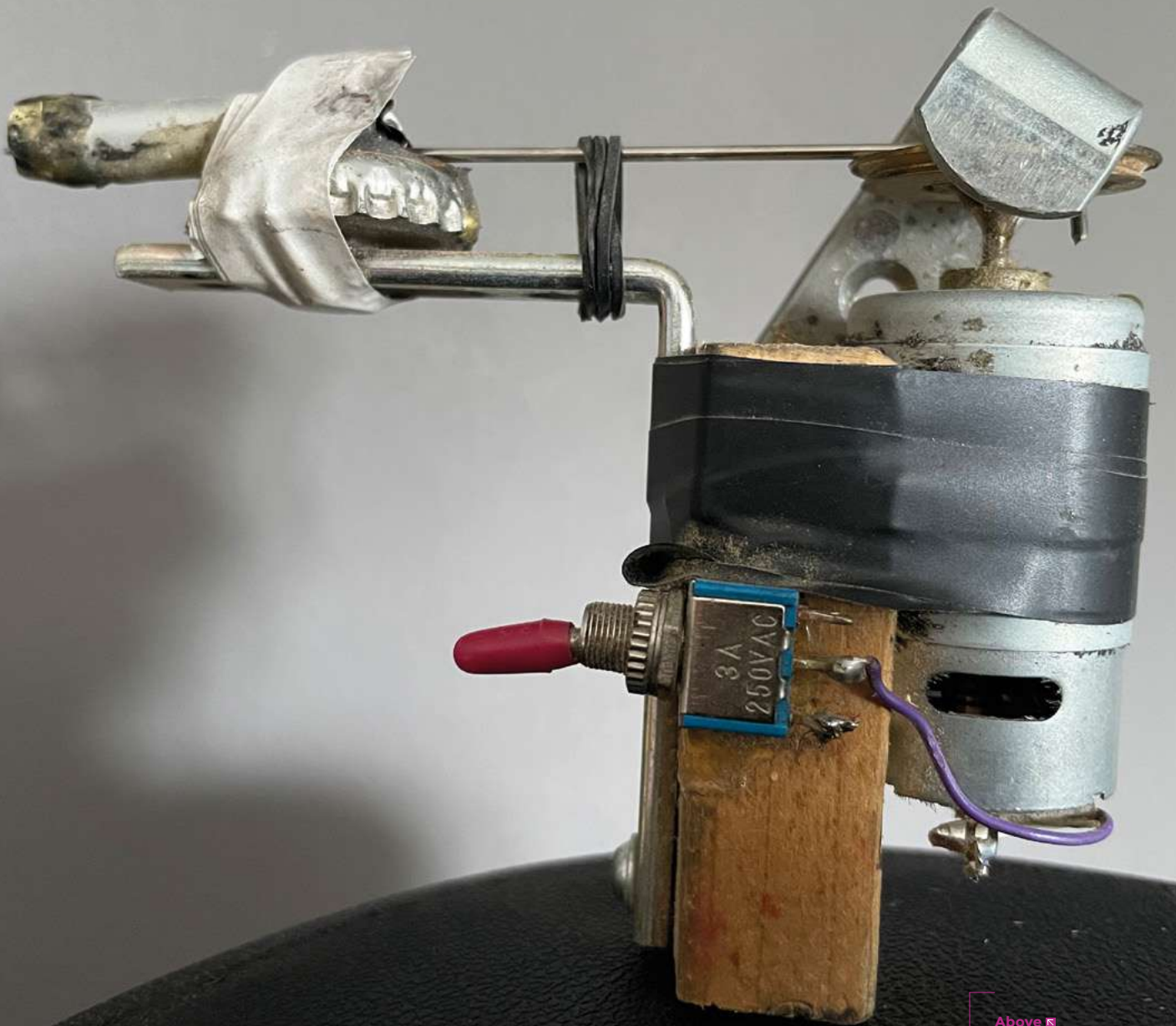
LK I was expecting it to break eventually. And, to be honest, I was kinda hoping for it because I wanted to do it again. And, this time, I'm actually hoping for it to get broken again because I want to do it again.

HS I may be making this up, but I'm sure you once mentioned that it's illegal to sell furniture in Germany unless you're registered. Is that right?

LK Yeah, it's a very broad description of this, but the craftsmanship in Germany is of a very high standard, right? At least we like to think so. So, if you want to be a carpenter, you're first an apprentice for three years or so, then you can be a carpenter and work under a master carpenter. If you want to educate other apprentices, or if you want to sell certain furniture, I think chairs is one of them, then you have to be a master. And it's the same for every field. I think the most plausible is electricians. If you are not a master electrician, you cannot, say, make a lamp and sell it.

But my interest is so general. I wanted to make lamps, but the notion of designing a lamp that's made out of wood and then obviously has electricity in it, it's just impossible.

I spoke to the TÜV and asked them, if I design a lamp and want to sell it in a >



Above
The tattoo machine
that started it all

store, how do I do it. And I would have to get it checked by their institution, which is a couple of hundred euros, and get a certificate. But I would have to do this for the next lamp design, and the next. And that makes them so expensive. I can't sell a lamp for 150 euros if it costs me more than that to get it checked. I'm not interested in mass production, I want to make one-off pieces.

I had already quit my job when I discovered this and remember having a big knot in my stomach thinking, 'what do I do?', and YouTube was the answer.

HS Could you not use YouTube as a way of selling lamps? It's not a lamp, it's a video prop?

LK Yeah, there are loopholes – this is not a lamp, this is art. But, when I quit my job to become a self-employed lamp seller, I really only quit my job because I hated working for other people, not because it was my dream to sell furniture and lamps. I didn't know YouTube really existed as a thing for me, and once I figured out people were actually making money off this, I was like, OK, I need to get a camera, I need to give this a try. Because that would be better than building stuff to sell it. I wasn't interested in selling stuff. I don't want clients. I don't want that pressure from anyone else except me, so YouTube worked out perfectly for me.

Below ↓

All good workshops need a dog



HS The job you quit was as a Display Artist for Urban Outfitters, if I remember correctly? Designing displays within a store. That sounded like a brilliant job.

LK It was. It was a great job, but it wasn't for me. It was probably the perfect job, but I am not a good employee. I was asked a couple of years ago if I would do a talk about my career and how I made this job for myself and followed my dreams, blah, blah. I don't like 'follow your dreams'. It was the other way around. I avoided my nightmares. That's how I got here. I never dreamed of this, I didn't know this existed. So, I think avoiding your nightmares is much more efficient than following your dreams.

HS With your design school background, when you create something, how much of that project is art over functionality? Dovetails versus pocket holes for instance.

LK It's more, and this is hard to explain, but I have this internal measuring unit of how much work should go into a project. I know how much time I can put into a project, and there's this bucket of work I've put into it, and depending on how full the bucket is determines how the project looks and whether I use pocket holes or dovetails, for example.

HS You work a lot with wood and with metal, as well as a few other materials, all of which require their own set of skills. Where have you learned all your techniques?

LK All YouTube. That's the cool thing. It's all full circle. There are some things – I had a couple of jobs where I learned some skills. I worked as a flight case builder for three years, just filling those black flight cases. Which sounds very, very trivial. It's not though, it's crazy. You have to work so precisely, otherwise, the catches won't close and all these things, and everything is building boxes. So I learned a bunch of stuff there. It was my *Karate Kid* apprenticeship. But a lot of it is YouTube.

I remember watching Jimmy DiResta – I saw his TV show online, and then I watched a bunch of his videos without realising it was the same guy. Eventually, I noticed he had a weekly schedule and a podcast, and it was all exactly what I needed to see and hear. Right when I quit my job and I couldn't sell lamps, there were these people telling me that they do this for a living. It was perfect timing. I feel like I'm the second generation YouTuber and they're the first.

HS As well as those makers, what else influences your work?

LK I like to listen to a lot of hip-hop, like super-aggressive hip-hop that is the complete opposite of me and has nothing to do with my world. And I like to watch horror movies, super-scary and bloody horror movies. I like to explore the opposite of what I have. A view into a completely different world. The Fantasy Filmfest is a huge inspiration for me. These movies that go right to DVD; they don't go into the big theatres. I like to think about how they got made? How did they think of that? That's the biggest inspiration. And, with hip-hop, the personas, and why they feel like they do, and how do they come up with those lines. They're in their own universe, they have their own rules. I just love that. It's how I feel when I'm building stuff. I'm telling myself a story that I don't know the ending of. I don't like to make sketches, I don't like to know if it works. If I see someone else had the idea and did a full video about it, I don't even want to do the idea anymore. I want to have that unknown. This is the idea, this is the stuff you have, now try to make it happen.

HS Is there anything still on the list? Projects you still want to work on?

LK I don't know if you saw it on Instagram, but I bought a Multicar. It's so good. It's the slowest car ever; it is painfully slow – 45 kilometres an hour and that's it. But it has torque; you can tow pretty much everything with it. So

my plan is to take the world's smallest pub that I built a couple months ago and put it on at the back of the Multicar.

Something is holding me back at the moment, though. I have all the parts, I should be able to do it, but I don't know what it is. I experience that quite often – I have an idea, and everything should be good to go, but I'm not doing it. And then, eventually, it turns out I wasn't sure about the colour, or something else that was missing that I didn't know at the time. So I don't push it. But that's the project I'm looking forward to.

HS Do you think you'll ever just get to the point where you're going to stop doing weekly videos? Or is this you for life?

LK I don't know. Like, that's the one thing that I'm really scared of, like, what happens when I get sick? Because at the beginning of the year, I hired my best friend. So now we're both relying on my mental and physical health. So I think it's a good idea to broaden stuff and have more income streams. I love doing the TV stuff [Laura recently started presenting a new TV show], because whenever I'm working with the TV people, I think, like, oh man, I love YouTube. And, when I do too much YouTube, I start really looking forward to working with actual professionals again. It's a cool balance. I kind of hope that I can keep doing this. You know, I think it's really cool. And as I said, there's no other place for me. Where would I go?

HS You have YouTube, you have TV, you have your podcast, and you sell merchandise. Is there anything left?

LK I think I would like to actually have a couple of products now. Some of the furniture I'm building, if you look at them in a different context to 'this is just what I built this week' and is only the product of seven days' worth of work, I think some of those ideas aren't that bad. And if you put some more work into them, they could be pieces that would sell. But I



Left It might not look like it now, but this will become a pub on wheels

would want someone who takes the prototypes and does the whole production for me. I'm not interested in all that. But I think it would be cool to have a line of plywood furniture.

HS So we won't be seeing a run of the Laura Kampf bench across Köln?

LK A newspaper interviewed me about the bench. And for the interview, they also approached the city saying, hey, wouldn't you want to work with her? And you know, maybe collaborate on this because this might be a cool thing. And they said that they don't have the personnel. But honestly, if they would have done it, that would have made it so boring. Working with somebody in an office telling me where the broken benches are so I can go and fix them. That's a job.

HS Is there anything you've ever made that you haven't wanted to share? A build just for you?

LK Until I hit publish, I feel like that every week. It feels like I'm just making it for myself. I talk very positively about YouTube, and that's genuinely how I feel about it, but sometimes it's really hard on me because I'll work seven days on a video, think it's the best thing I ever did, and it makes me so happy. I'll edit it for hours, sink all this time into it, all this

energy, and then the video tanks, and it kinda ruins it for me. I'm in a super-good mood right now because the bench video did so well, and people understand what I'm trying to say. But, there are other cases where it doesn't work as well, and where I feel like I've dropped the ball and couldn't get my excitement across. And that's always super-disappointing because I'm always excited about the stuff I make; I always have some angle I find super-interesting, otherwise, I'm not motivated to do it. And, when the video tanks, it makes me feel like I lost the opportunity to spread that excitement, to spread that motivation, and that feels like I wasted my time. And that's the downside of YouTube.

I mean, I think every creator takes it in a different way. And you need to find a way to deal with this, and it's really important to talk about it. This is my dream job and I can do whatever I want to do as long as I don't drop the ball. I hired my friend, so now I can't drop the ball for the both of us.

Laura produces a video every Sunday on her YouTube channel (hsmag.cc/LauraKampf). You can also follow her on Instagram (instagram.com/laura_kampf) and, for any German-speaking readers, her podcast – Raabe & Kampf – with friend and journalist Melanie Raabe can be found wherever you listen to podcasts.

CODE
THE
CLASSICS
VOLUME 1

CODE THE CLASSICS VOLUME 1



Brimble
Crookes
Gillett
Malone
Tracey
Upton



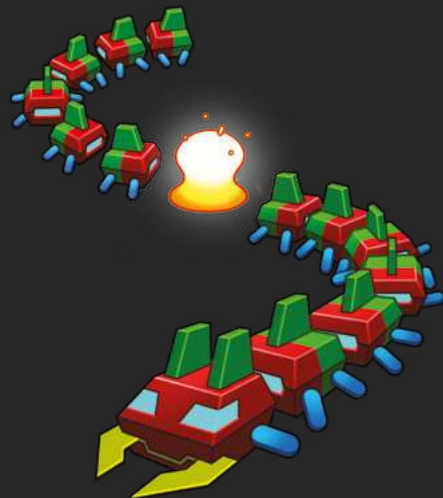


CODE THE CLASSICS VOLUME 1

This stunning 224-page hardback book not only tells the stories of some of the seminal video games of the 1970s and 1980s, but shows you how to create your own games inspired by them using Python and Pygame Zero, following examples programmed by Raspberry Pi founder Eben Upton.



- *Get game design tips and tricks from the masters*
- *Explore the code listing and find out how they work*
- *Download and play game examples by Eben Upton*
- *Learn how to code your own games with Pygame Zero*



Available now hsmag.cc/store

PLASTIC MILK CARTONS



Even recyclable plastic is made from non-renewable materials. Rosie Hattersley suggests simple ways to reuse it



Rosie Hattersley

[@RosieHattersley](#)

Rosie Hattersley writes tech, craft, and life hacks and tweets [@RosieHattersley](#).

W

e're all familiar with the environmental mantra reduce, reuse, recycle, yet our seas and landfill are crammed with more packaging and plastic than ever before. Ocean Conservancy,

which campaigns for trash-free seas, warns that we are just not doing enough to tackle plastic pollution. Uncollected plastic bottles that could easily be recycled account for the greatest amount of such pollution, so focusing on improving how much is collected could have a significant impact. Nonetheless, recycling is the least desirable of the aforementioned three Rs: reducing the amount of plastic we produce in the first place, and reusing plastic items wherever possible do far more good.

The clear plastic that water bottles are made from (polyethylene terephthalate or PET) was first manufactured by DuPont in 1921 as a textiles manufacturing by-product, with the convenient drinks container version of PET debuting 20 years later. Its water bottle guise now accounts for 30% of all PET and polyester produced each year. Unfortunately, discarded drinks bottles account for a similar proportion of non-biodegradable waste.

Plastic milk bottles are among the easiest and most commonly recycled items of domestic plastic, partly because the dairy industry specifies design similarities and has a unified approach to reclaiming and recycling plastic. In the UK, bottles are made from food-grade HDPE (high-density polyethylene) and are stamped with the number 2 to show they are both recyclable and are partially made from recycled material. In fact, up to 40% of a milk carton is made from recycled plastic. It's not quite a closed loop, though: because they are made from coloured plastic, milk bottles can be recycled into other items, but not new milk bottles.

When it comes to recycling projects, the white plastic also rules out some – but not all – ideas in which light is intended to permeate/penetrate through groups of bottles. However, lanterns and lampshades with pieces artfully cut out can work well. Since milk bottles are made from noticeably thinner plastic than many food-grade storage products, it is easier to manipulate, mould, and cut. This means there's a greater range of upcycling projects it can be used for – from decorative finishes on a child's fancy dress outfit to the centrepiece of an eco home.

FUNKY PLANT HOLDERS



One of the most straightforward ways to reuse one-pint plastic bottles is as receptacles for herbs or other small plants. Drill circular holes for several

along the length of a shelf, or place them at the edge of a garden bed as an easy herb garden. Larger-capacity bottles lend themselves to more decorative uses, such as showing off your beautiful trailing blooms. The CraftBits version shown here, with the trailing plants resembling tresses, is one of many such examples you'll find online.

To make your own, first cut off the base. Invert your plastic milk or juice carton and note how, with the handle facing you, it looks a little like a nose. Squint a bit to see whether your bottle is calling out to be made over as Frida Kahlo, Amy Winehouse, or another iconic lady, then accentuate the carton's face-like qualities by applying some brightly coloured paints and any accessories that seem to fit. Once the paint has dried, pierce holes near the top of the cut-off container to string up your planter, and then add soil and pot your plant. →

Project Maker
SHELLIE WILSON

Project Link
hsmag.cc/CartonPlanter

Left ♦
Inverted plastic milk bottles can be made over to become glamorous decorative planters



FEATURE

BOTTLE WALLS AND HOMES

Project Maker
**MORIMOTO
RESTAURANT**


Project Link
hsmag.cc/Morimoto

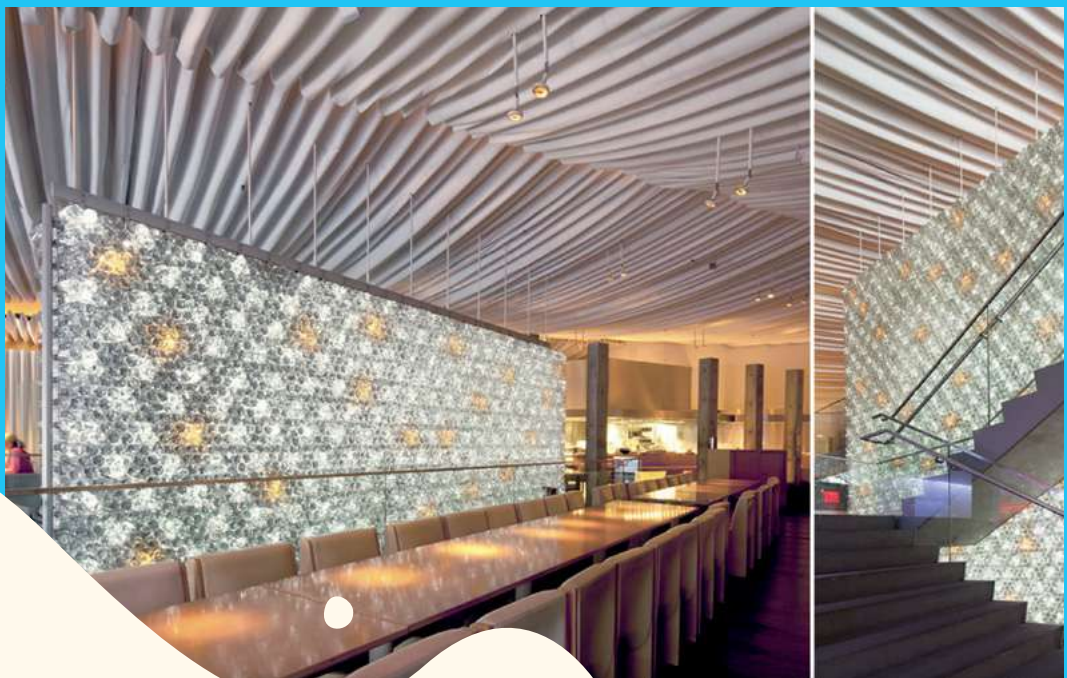
Permaculturalists have recently embraced the glass bottle walls beloved by enterprising 1970s earth-conscious homemakers, but there's also a sizable movement devoted to plastic bottle homebuilding.

Made from discarded plastic bottles, these mostly take the form of walls filled with earth and 'windows' glued together but left unfilled. The now-defunct Morimoto Restaurant in New York was a celebrated example – so much so that its eco wall was its unique selling point. The backlit two-storey PET water bottle wall formed a visual

link between the restaurant's two levels and was held in place with the help of a steel frame from which the bottles were suspended, plus steel rods that helped keep the plastic bottle wall's form. The particular brand of spring-water bottles was chosen specifically for their curves, distinctive bases – the parts on show – and the visual effects that could be achieved.

Websites such as instead.com and permaculture.co.uk offer insights into some of the ways that plastic bottles can be used for more practical wall-building projects, including vertical vegetable gardens: hsmag.cc/VegGarden.

Right 
Bottles are a cheap and versatile building material



STORMTROOPER HELMET



You'll need two gallon-size plastic milk bottles for this project, which means a fair intake of cow juice, but your resident primary

age *Star Wars* fan will appreciate the results. The white plastic lends itself nicely to this design, and can easily be cut to shape using a craft knife or kitchen scissors. You'll need to measure your stormtrooper's head to get the measurements right before painting and embellishing the helmet. Cut away the base and most of the sides of one plastic bottle for



the horizontal part of the helmet that will frame the face. Then use a semicircular piece from the upper part of the second bottle, plus a disc for the top. Glue the three pieces together at right angles to each other, then check the sizing on your model's head so you can work out where to place the eyes. There's every chance the decorations you add will disguise these once the helmet is being worn. Use photos of *Star Wars* stormtroopers as a guide to

how to decorate the helmet's exterior, and dress your child to fit the space combat theme.

Project Maker
FILTH
WIZARDRY

Project Link
hsmag.cc/CartonHelmet

Left ♦
Two large bottles provide the basis of an iconic *Stars Wars* look

MILK CARTON LAMPS

This IQ Lamp Instructable came about after maker Edwardo had seen similar such projects using other materials, and decided to experiment with reusing milk bottle plastic to build

his own after realising that a similar one bought online was less robust plastic than he'd hoped. Using roughly 15 two-litre milk bottles and some LEDs, such as those used for Christmas decorations, he was able to come up with a template for his own upcycled illumination. He printed out the lamp's 30 interlocking pieces templates on paper, cut them out and, using sticky tape, stuck them to the plastic (since marker pen ink is often repelled by plastic) to ensure accurate shapes. A sturdy pair of scissors was strong enough to cut through the plastic. Since the LEDs are low-power and generate little heat, they're safe enough to use as the light source, even wrapped around a stress ball, as in this project. An optional

colour-changing keyring can also be used, but since Edwardo published his Instructable, colour-changing LED string lights have become readily available. Punch holes for the plate hooks that keep the lamp's plastic form in place, and string it up in place. □



Project Maker
EDUARDO LEON

Project Link
hsmag.cc/IQLamp

Left ♦
IQ lamps provide a construction puzzle and make a great upcycling project

IN THE WORKSHOP: knife handling

By Ben Everard

Giving new life to old tools

W

e have a set of kitchen knives where the steel of the blades are still in good condition, but the handles are starting to show their age. In some

cases, they're a bit tatty. In

others, they've been burned where they've been left too close to the oven. There's no point in throwing them away when the steel is still in good condition, so I decide to re-handle them.

Fortunately, the knives in question are 'full-tang'. That means that the steel from the blade continues all the way through the knife-handle. The handle isn't much more than a block of 'something' attached to the outside of the 'tang'.

You can make knife-handles out of lots of different materials, and there are plenty of shops online dedicated to just these 'scales' (as handle material is known in the knife-making world). You can get resins (often imbued with colours, glitters, or swirls), exotic woods with interesting grains, and a host of other things. However, I decided to use a chunk of oak off-cuts. This is partly because I'm cheap, and partly because – OK, it's entirely because I'm cheap. Knife scales are expensive, and I want to do a set of knives. I may live to regret this decision, but so far it's working out OK.

I'll attach the wood to the tang with epoxy and then shape it using the tang as a guide. But there's one part of the handle that can't be shaped after it's attached,



Right ♦
The finished knife – ready to get back to work chopping vegetables



Left ♦ Only the front part of the handle has to be lined up and shaped before gluing

Below ♦ The tang provides a guide you can shape the handle around

and that's the leading edge – the part at the front, perpendicular to the blade. The first task, then, is to take the angle of this down from 90 degrees to a more aesthetically pleasing 75 or 80 (I didn't use a protractor, and just did this by eye).

There are many ways of shaping wood, and throughout this process I used 'power carving'. In this case, I used a cylindrical sanding bit on a rotary tool to remove the bits I didn't want. There's no need to worry about the corners here. The basic idea is to have blocks of wood that are too large, with just the front face shaped. I'll remove the excess material after the handles are glued on.

Most knife tangs have holes in, and you can put a brass bar through this. In my experience, this isn't necessary for strength – epoxy is strong enough on its own to hold the handles in place. It's a purely aesthetic thing. I decided not to add them – mostly because I have a lot of knives to do and not much time to do it in.

Gluing up is just a case of roughing the surface of the tang with some coarse sandpaper, mixing the epoxy, applying it, popping the handles on, and clamping up. The only thing to watch out for here is to make sure that you get the front face right. Too much epoxy and it will squidge out too much, and it's hard to remove from the knife-blade. Too little and the very front bit won't be glued (food might get stuck in here). I aimed to get a tiny little bit to squeeze out – so little that it wasn't noticeable when looking at it. I think I succeeded, but I'll only know for sure after I've been using it for a while. If things start to get caught under there, then it's too little.

At this point, I had big square blocks of wood attached to the front of the knife. Shaping the handles seems like it should be a tricky process, but actually, it's very easy. I used three tools: an angle grinder with sanding flap-disc for very rough shaping, a rotary tool with a cylindrical sanding bit for slightly more detailed work, and a sanding block for the more fine work. Getting the profile of the handle is easy because all you



have to do is go in with the angle grinder perpendicular to the tang, and keep removing wood until you reach the tang.

The angle grinder with a flap disc is a powerful beast. It's great for power carving because it rips out large chunks of wood quickly, and in this case, isn't damaged when it hits the metal of the tang. It does produce a lot of dust, so a good ventilator is essential.

At this point, I had shaped the handle in two dimensions, but it was still square and blocky in the third. Time to smooth round the top of the handle. I did this with the rotary tool. Doing this free-hand isn't too hard, but you must be careful not to remove too much in the wrong places. I found that playing with the speed of the cutter until I got the right level of aggressiveness was essential to keeping control of the tool.

Once that was done, I ditched the power tools and went at it by hand. First, with the sanding block, I ensured everything that should be flat was flat, and made minor adjustments to corners. Then, I went through the grades of sandpaper to get to a nice smooth finish. Finally, a couple of coats of food-grade wood-oil finished off the knife.

I'm really pleased with how it looks, but more than this, I like the feel of it in my hand. It's hard to say exactly what is different about it, but it's just nicer to use. Unfortunately, I still have four more knives to re-handle. Back to the workshop! ▣

SUBSCRIPTION

SUBSCRIBE TODAY

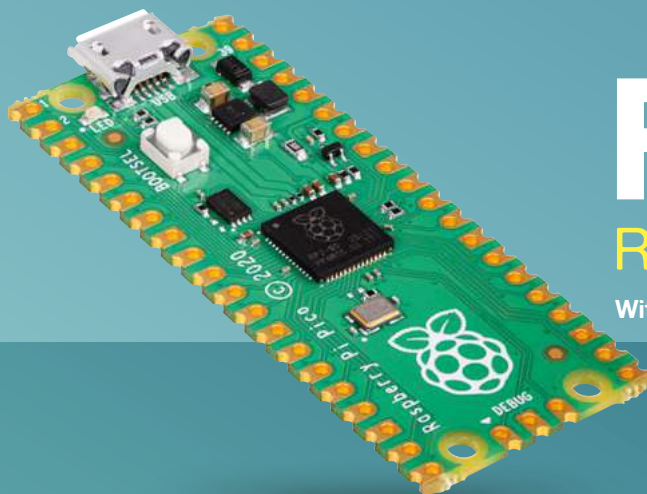
Get 12 issues of HackSpace magazine
delivered to your door for just

£55 (UK)

£90 (USA)

£80 (EU)

£90 (Rest of World)



FREE!

Raspberry Pi Pico

With your first 12-month print subscription

This is a limited offer. Not included with renewals.
Offer subject to change or withdrawal at any time.

 **Subscribe online:** hsmag.cc/subscribe

Subscription starts with next issue

HackSpace
TECHNOLOGY IN YOUR HANDS

FORGE

HACK | MAKE | BUILD | CREATE

Improve your skills, learn something new, or just have fun tinkering – we hope you enjoy these hand-picked projects

PG
72



CNC UPGRADES

Cut quicker and easier with these essential add-ons

PG
76

ARCADE

Give your arcade cabinet a vinyl makeover



PG
82

FREECAD

Create organic-looking shapes with curves and extrusions

PG
88



SIMPLE GAMES

Build a game with just two buttons

PG
96

NIBBLE

Get started with this DIY hand-held games console

PG
70

SCHOOL OF MAKING

Start your journey to craftsmanship with these essential skills

70 Remote control Pico

Remote control Pico

Control your microcontroller from your PC



Ben Everard

[@ben_everard](#)

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

Raspberry Pi Pico is a microcontroller. That means that it's a processing unit designed to run on its own without a

keyboard and monitor. However, sometimes it'd be useful for it to be a little less independent. Obviously, you need to hook it up to a computer to program it, but it can also be useful to control Pico from a computer to test out a circuit (either one with an embedded Pico, or using Pico as a tester for something else), to set it up with some initial values, or to just create a programming environment using a text editor that doesn't support MicroPython out of the box.

You can use the `mpremote` command to control Pico. This can mean sending data, running code, or accessing the file system. Let's take a look.

First, you'll need to install `mpremote`. You'll need Python (python.org) installed on your computer; once this is there, you can get `mpremote` with:

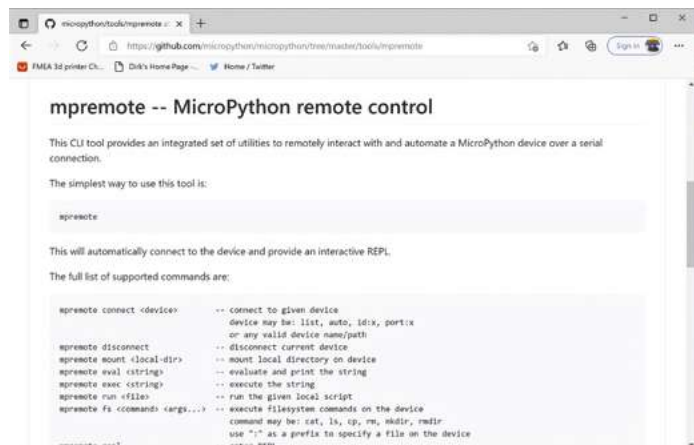
```
pip install mpremote
```

You'll need the very latest version of MicroPython (1.16), so grab this from hsmag.cc/picomp and flash it to your Pico (the process is described on that page, if you haven't done it before).

You're now ready to start controlling Pico from your computer. Open a terminal (or command prompt if you're on Windows), plug Pico into your computer, and run the following to check everything's working:

```
mpremote exec "from machine import Pin; led=Pin(25, Pin.OUT); led.value(1)"
```

This should connect to Pico and turn the built-in LED on.



Using the `exec` command, `mpremote` can run one or more MicroPython commands (separated by semicolons). This doesn't maintain state between different calls, so if you create a variable or import a module, it won't still be there the next time you run `mpremote`.

If you've got a set of commands that you run regularly, you can create a config file that links them to short cuts. There are a few included in `mpremote` by default, including:

```
mpremote setrtc  
mpremote df
```

THONNY

You can do most of the things `mpremote` can do in a graphical way with a MicroPython-specific IDE such as Thonny. With this, you can run files stored on your computer on Pico, and interrogate the file system. Depending on what you're trying to do, it may be easier to use this.

However, if you're trying to interact with Pico from a program, or perform the same task multiple times (such as setting up a lot of Picos), then `mpremote` will be easier.

Above

The best source of documentation is the project GitHub at: hsmag.cc/mpremotegit


```

Command Prompt
Microsoft Windows [Version 10.0.19041.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ben>mpremote connect list
COM10 E660081007987731 0005:2e8a Microsoft None

C:\Users\ben>mpremote connect COM10 ls
ls :
    0 Adafruit_Bus_Device/
    2011 DebouncerPIO.py
    2058 DebouncerPIO.py
    2400 PIOBeep-harmonic.py
    1859 PIOBeep.py
    2438 PIOBeep_harmonic.py
    15705 adafruit_bmp280.py
    0 adafruit_bus_device/
    152 fake_time.py
    392 time.py

C:\Users\ben>mpremote connect COM10 df
mount size used avail use%
1441792 94208 1347584 6
Adafruit_Bus_Device 1441792 94208 1347584 6
adafruit_bus_device 1441792 94208 1347584 6

C:\Users\ben>

```

Left ♦
It's probably time to clear out this Pico as it's been filling up with libraries from the different projects we've been working on

The former of these will set the RTC on Pico to the current time. Unless you have a battery backup, this won't carry over if you power down Pico, but can be useful for constantly powered projects.

The second of these shows how much space is left on your device's file system. You can expand these defaults to run any MicroPython code you like by creating a file: **.config/mpremote/config.py**. There are full details of the format for this file on the GitHub page at hsmag.cc/mpremote.

FILING AWAY

Running code is useful, but it's also incredibly useful to be able to access storage. There are a couple of ways mpremote works with the internal storage of your device. The first is to let you run file system commands (the available commands are: cat, ls, cp, rm, mkdir, rmdir). For example, to list the files on Pico, run:

```
mpremote ls
```

One potentially powerful way of working with data on Pico is to mount a directory from your main computer to a location on Pico.

For example, if you enter:

```
mpremote mount .
```

This will mount the directory your terminal is in to the location **/remote** on Pico and drop you into the MicroPython interpreter where you can run commands. If you enter:

```

>>> file = open("/remote/Testfile.txt", "w")
>>> file.write("Hello from MicroPython")
>>> file.close()

```

You'll then find a file called **TestFile.txt** in your local directory with the contents "Hello from

MULTIPLE PICOS

In the examples here, we've been assuming that you have just a single Pico plugged into your computer. This may not always be the case, and mpremote can connect via port or board ID.

If you run:

```
mpremote connect list
```

...you can see the available MicroPython devices.

You can then connect with them, for example, with:

```
mpremote connect COM9
```

This will connect to the device on COM9 (you'll need to change this depending on which port your Pico is on) and open the MicroPython interpreter interface.

You can run a specific command on a specific Pico with, for example:

```

mpremote connect COM9 exec "from machine
import Pin; led=Pin(25, Pin.OUT); led.
value(1)"

```

MicroPython"). You can combine the mount option with others, such as exec, and run to run code automatically that sends data to a computer. This is useful if you want to use Pico to interface with some sensors and save the data, as it will let you store more than you can fit on Pico's storage. It's not a great long-term solution (use an SD card for that), but if you want to quickly grab a lot of data, it could be a great option. Getting training data for a machine learning model could be a great use case for this. □

Upgrade your CNC

We test out some CNC 3018 Pro modifications to make it safer and cut better



Jo Hinchliffe

[@concreted0g](#)

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!




ur CNC 3018 Pro worked out of the box, but it is a very bare-bones machine. Let's take a look at some modifications that make it a bit more pleasant to use.

Our first add-on is a 3D-printed tool holder. It's a simple print; we designed it in FreeCAD and uploaded it to Thingiverse: [hsmag.cc/ToolHolder](https://www.thingiverse.com/thing:458888). It is designed to hold the two small spanners for the collet nut and spindle, and it also holds two hex keys: one for general tightening of the machine and one that fits the spindle retaining bolt. We've designed it to have an oversize slot that fits over the side panels of the machine with a small piece of soft foam in the slot – this dampens the holder so that the tools don't rattle or shake off the panel. While we are talking about

3D printing, we have also printed some storage for our ER11 collet collection which allows us to use a range of different size tooling. This collet box is available to download and print here: [hsmag.cc/ColletBox](https://www.hsmag.cc/ColletBox).

Printing a range of work clamps is a great idea if you have access to a 3D printer. It's worth considering upping the amount of printed infill to make the clamps as tough as possible and, although we have used PLA prints successfully, perhaps considering printing in a tougher filament such as ABS or PETG. Again, if you aren't able to CAD something to your needs, there's a range of models to be found online.

We found that our CNC 3018 Pro was a bit prone to sliding around the desk, not so much during operation but more when clamping work or tightening the collet. If you don't need to move the machine too often,

Above  The small add-ons and modifications in this article make the CNC 3018 Pro slightly better than stock – and easier to use



YOU'LL NEED

- ◆ CNC 3018 Pro kit
- ◆ Z axis probe kit, or some wire and crocodile clips
- ◆ Push-to-make emergency stop button
- ◆ 2 × LM8LUU 45 mm linear bearings
- ◆ Short length of 12 V LED lighting strip

clamping/bolting it down to a surface would be the best option. We glued some scrap pieces of rubber to the feet, which made it much better (**Figure 1**).

The PCB on the CNC 3018 Pro is often referred to as a 'woodpecker' board, and it has numerous extra features broken out to header pins for accessories and add-ons. One thing it does not have, however, is a set of pins for an emergency stop or an emergency reset system. A true emergency stop system should be the obligatory big obvious button that, when hit, cuts all power to the system. This would involve wiring into the mains supply system and buying a suitably rated switch. We've gone for a different approach in that we have added an emergency reset button, which has the same functionality as the small reset button built onto the PCB. In fact, we have soldered onto the reset switch solder pads. With the emergency reset button pressed, the CNC 3018 Pro will stop movement in all axes and also the rotation of the spindle. We bought a budget emergency stop button that has two pairs of terminals inside: one for a normally closed (NC) operation and

one for normally open (NO). The reset switch on the PCB is NO, or 'push to make', so we soldered a wire to each side of the reset button and connected them to the NO terminals of the emergency stop button. The emergency stop button has a mechanism that, once pressed, will remain closed until you twist the switch/button cap to release it.

Testing the emergency reset system, we loaded a G-code file and zeroed the machine in a position to perform an air cut with no tool inserted. We let the →

LET IT SHINE

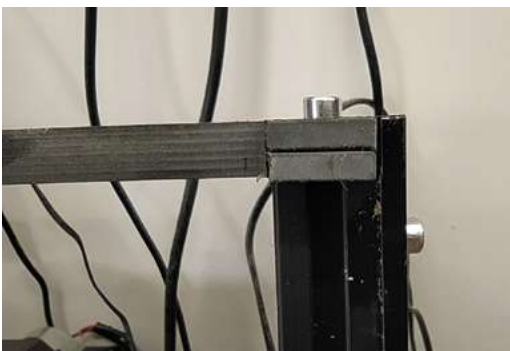


Our final modification, worthy of mention, is we realised that the small aluminium extrusions of the CNC 3018 Pro had slots perfectly sized to receive flexible 12 V LED lighting strips. While we could have gone all disco and kitted out every extrusion with RGB lighting, we went with the sensible option of adding a short length of white LEDs into the slot that faces the machine bed. It's a great addition that throws some light into an awkward spot that often is in shadow, and it increases visibility when changing tools.

Above ☒ Losing or wasting time looking for tools is no fun – this little tool-holder holds the important stuff that you will use regularly with the CNC 3018 Pro

Figure 1 ☒ Sometimes it's the simple things! Adding scrap rubber as feet stops the CNC 3018 Pro sliding on the desk

Below ◆ Our emergency stop button – available cheaply online – had both NO and NC switch terminals



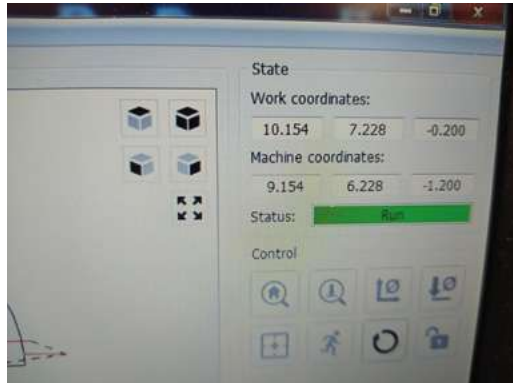


Figure 2 ■

With the emergency reset button pressed, it's worth taking a photo of the work positions so you can reset to the previous work zero, potentially saving a stopped job

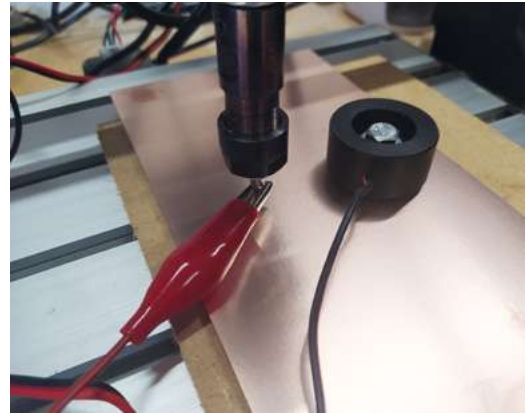
Figure 3 ■

Simple Z axis probe kits are available cheaply online, or alternatively, they are very easy to make

Figure 4 ◆

If you are using a conductive material, you can place your Z axis probe upside down and probe directly onto the material

machine air cut a little and then pressed the emergency button. It worked perfectly to stop both the spindle rotation and the axes movement. Releasing the emergency stop button completes the reset, and the machine co-ordinates zero in the position that it stopped in, and the work co-ordinates change. This is a pain if you wanted to perhaps swap a broken tool and then continue with the job. But, we have a solution! When you press the reset button and the machine stops, make a note, or take a photo on your phone, of the work co-ordinates. Once we reset the emergency stop switch, we can then reset the work position to these co-ordinates. In **Figure 2**, you can see we took a photo of the work co-ordinates, and we can simply send the following G-code to the machine to restore them: G92 X10.154 Y7.228 Z-0.200. Of course, then you can use the 'Return to safe position' button in Candle which we set up in last month's article to send a G0 X0 Y0 to return the spindle to the origin point of the disrupted job. Remember though, you may need to raise the Z axis before doing this. This emergency reset system is incredibly handy, and adds an additional layer of safety to the machine – definitely better than reaching over the machine to try and find the reset button!



One set of pins that are broken out on our board are labelled GND and A5. The A5 pin is commonly used in GRBL-equipped machines to attach a Z axis probe. Z axis probes are quite a simple arrangement consisting of two wires: one of the wires has a crocodile clip attached at the end, and the other is connected to a block with a metal touchpad area. The idea is that you clip the crocodile clip to the tool in the spindle, and place the block underneath on top of the work surface (**Figure 3**). Pressing the Z axis probe button in Candle, the Z axis is slowly jogged down until a circuit is made as the tool touches the block, the G-code then moves the Z axis off slightly and repeats a very small probe at a slower speed to increase the accuracy of when the probe/tool tip touches. The important part is that you accurately measure the height of your probe block – using digital callipers should be accurate enough for most uses, but you might consider using a micrometer set if you want to get really accurate. You can then send a G92 ZXXX command to GRBL, where you replace XXXX with the height of your probe block.

You can make a simple Z axis probe out of a pair of wires, a crocodile clip, and something conductive as your probe block. A common choice, especially on small machines, is to use a small piece of copper-clad PCB. As a final thought, if your material is conductive, you might be able to Z probe straight onto it, or even place your probe block upside down on the material's surface to turn the surface into a conductive probe area (**Figure 4**).

So far, our CNC 3018 Pro hacks have been either additions, electronic additions, or software tweaks. We have, however, performed one mechanical modification to our CNC 3018 Pro to try and reduce some of the flex in the Z axis. We noticed that some of the flex in the Z axis assembly was play in the linear bearings on the short vertical Z axis rails. Pulling it apart for inspection, we discovered that each rail had two short linear bearings sat within the housing, and a small gap of around 4mm in



the middle. We imagined that if we increased the contact area, and also perhaps used some slightly higher-quality linear bearings, we might remove some of the flex.

The Z axis spindle housing measures 34mm deep on our machine – we couldn't find any 8mm bearings that would fit, but we did see that 45mm 'LM8LUU' bearings are commonly sold and used a lot in 3D printer designs (Figure 5). 45mm is too long, so if you undertake this modification, you will lose a small amount of travel in the Z axis. The CNC 3018 Pro is a small machine and, therefore, we feel we are unlikely to cut deep objects or parts on it, so we were happy to try and improve the Z axis play. Stripping the Z axis, we began by removing the bolts holding the X axis rails and then loosened the grub screws, retaining the threaded rod to the X axis stepper motor. We then found that if we removed the side bolts that hold the side panels to the X-directional aluminium extrusions, and then loosened the lower side panel bolts, we could slide the assembly off.

Next, we undid the upper and lower Z axis stepper motor screws and hex key bolts. We actually realised that we only needed to remove the more awkward-to-get-to bolts that come up through the Z axis assembly through the white nylon standoffs. There are access holes through the spindle clamp housing to allow you to reach these bolts, albeit with a very long hex key. Once the stepper motor is loose, undoing the grub screws on the coupler will allow you to slide the spindle housing and the threaded rod back and forth on the rails. We only partially removed the Z axis rails – we did this by using a small punch and tapping the rails out towards the stepper motor end of the assembly. Once we had

enough space to bring the spindle housing off the rails, we did so. Similarly, we used our small punch to tap out the small pairs of linear bearings that our machine had come supplied with before tapping in the longer bearings. We shared the excess length of the new bearings over each end of the spindle housing, but you may prefer to leave the excess all at the top to keep the maximum depth of cut available. Once fitted, we found that they really did minimise the flex on the Z axis rails, with any remaining flex being the X axis rails bending slightly, moving the entire Z axis assembly. Definitely an improvement and worth the effort! ☐

QUICK TIP

We have the crimps to make DuPont/header-style connectors for our wiring. If you don't have them, you could sacrifice breadboard cables or solder to header sockets for your cable connectors.

NO LIMITS?

A popular modification to the CNC 3018 Pro that we didn't do is to add limit switches. Limit switches are simply microswitches attached to the CNC on the XYZ axes at the furthest possible points of travel before that axis crashes. There are numerous examples online of attaching them in position, which range from directly bolting them to parts of the machine, through to adding 3D-printed brackets. Limit switches are useful in that they can allow you to home the machine, which might make it easier if you are cutting projects that are close to the size limit of the bed. Limit switches will also stop the machine travel just at the point before the axis crashes – this is useful if you accidentally set the machine to jog a too-long distance, for example. One thing that people think limit switches might do is save a job if the job is too large or overlaps the edge of the machine travel – it won't! If your job doesn't fit and the machine stops, there is no real way to reposition the job and continue. In some ways, it's a better approach to make sure you know the size of your cutting job and work methodically to reduce the chances of crashing.

If you add limit switches, you need to enable limit switches in GRBL, which is achieved by sending a \$21=1 message. Similarly, you may need to send a \$22=1 message to enable homing. While researching limit switches, we read that some PCBs on CNC 3018 Pro machines have the pins reversed, so make sure to run plenty of tests!

Figure 5 ♦

Requiring some disassembly of the CNC 3018 Pro, we found adding the larger bearings to the Z axis removed some of the inherent flex in the machine



Below ♦

The larger bearings installed, ready for the Z axis to be reassembled to the machine



Build an arcade machine: Decorate your cabinet



K.G. Orphanides

K.G. is a writer, maker of odd games, and software preservation enthusiast. Their household can now hold very retro Street Fighter II tournaments, and that's beautiful

@KGOOrphanides

You've built an arcade cabinet, but vinyl decals and edge moulding will bring it to life

Most arcade cabinet kit suppliers print pre-designed or custom vinyl decals to decorate your cabinet. Third-party printers can produce vinyls to your specification, but make sure that you provide accurate measurements.

Our vinyl decals, bought from Omniretro (magpi.cc/omniretro), arrived on a roll and had to be cut out, but some firms will die-cut vinyls for you. We'll use a wet application process, which makes it easier to remove and reposition decals for a short while after initial placement, to help you get a perfect alignment.

01 Flatten your vinyl decals

If your vinyls all came on a single roll, the first step is to cut each of them out. First separate them, if they're on a single roll, but leave generous margins. Spread them out on a table or on the floor and weigh them down – coffee table books and textbooks are good for this. Leave them for at least an hour or two: 24 hours is better.

02 Cutting out

Now they're flat, it's time to cut out your vinyls. Try to get rid of all white matter on straight edges. The easiest way is to line up a long metal ruler so that it just covers the edge of the printing, and run a scalpel down the outside of it. Curved sections for the cabinet side panels are trickier, but you don't need to worry about these as they're easy to trim down once fitted. For now, trim them freehand and leave as much white overmatter as you feel comfortable with.

03 Partial disassembly

Depending on the design of your cabinet, you may need to remove a side panel to take out the acrylic marquee and screen panels. Before doing this, use a liquid chalk pen and ruler to mark the edges of your LCD display on the acrylic, so we can accurately hide the bezel.

If you've previously fitted joysticks and buttons to your control panel, this is the time to remove them too. Apply steady pressure to the rear of snap-in style buttons to pop them out of the cabinet. People with large fingers may find a ButterCade Snap Out Tool useful for this.



▲ Mark up in chalk pen and use a metal ruler to help cut your screen decal to size

04 Applying vinyl to your marquee acrylic

Two acrylic parts require individual application of vinyls: the marquee and the screen that goes in front of your monitor. The former is easy: remove the backing from the vinyl marquee decal and any protective film from the acrylic. Spray both the acrylic and the adhesive back of the vinyl with two or three squirts of application fluid. You want them to be damp all over but not awash.

Pick up the vinyl decal in both hands and, starting at one end of the acrylic, line it up with the edges and paste it down. If you're not happy with the positioning, firmly hold the vinyl and snap it back up – the application fluid will help it release easily.

Once it's positioned, use your applicator and a cloth to smooth it down, drive out any excess water, and remove any trapped air bubbles under the vinyl. Trim any excess vinyl spilling off the edge of the acrylic with a knife.

05 Measuring your screen acrylic

Cutting your screen decal to size is awkward. Before removing the screen acrylic from the cab, we marked the inner position of our monitor's bezel on the acrylic using a chalk pen. If your cabinet has a detachable VESA mount, bring the monitor with you to help line everything up.

“ Grab your screen vinyl and mark up the area to cut out ”

Measure the distance between the edge of the acrylic and the chalk line you drew on it. Measure in multiple places to be sure of distances. Our 24-inch monitor's positioning and bezel size means that we cut 35 mm in at the top and sides, and 65 mm from the bottom – yours will differ.

06 Cutting your screen decal

Once you've taken the measurements, grab your screen vinyl and mark up the area to cut out. Mark on the side showing the picture, paying particular care to the corner positions. Double-check these by placing the acrylic on top to make sure both sets of marks line up. →



Build an arcade machine: Decorate your cabinet

TUTORIAL



- ▶ Use a vinyl applicator and a cloth to stick down, remove excess moisture, and eliminate air bubbles from your decals

You'll Need

- ▶ Vinyl decals
 - ▶ U-moulding/
T-moulding
 - ▶ Scalpels/craft
knives
 - ▶ Strong scissors
 - ▶ Liquid chalk
marker pen
 - ▶ Metal rulers, tape
measures
 - ▶ Vinyl application
fluid
 - ▶ Vinyl applicator
 - ▶ Neoprene glue
-
- ▶ We marked the inner position of our monitor's bezel on the acrylic using a chalk pen

Top Tip

Screen materials

Acrylic scratches really easily, so tinted tempered glass is an excellent alternative for your cabinet screen.

Grab your metal ruler, place it along your marked line, and cut a rectangle out of the middle of the vinyl decal with a blade. If in doubt, err towards leaving too much vinyl rather than too little. To check positioning, put the acrylic over your monitor, and your vinyl over the acrylic: they should all line up.

07 Screen decal application

Now, turn the vinyl upside down, remove its backing, spray it and the acrylic with the application solution, and stick it down using an applicator and cloth. Residual chalk marks can be wiped off using a bit more of the application solution.

Allow both the marquee and screen decals to dry for a day, trim them if needed, slide them back into your cabinet, and reattach anything you removed. This will probably be the last time

you do this, so make sure the side panels are on securely and are correctly lined up and bolted to your stand, if you have one.

If you plan on back-lighting your marquee, this is a good time to put in your light. We used adhesive tape and supplied clips to mount a 50 cm USB-powered LED light on the underside of the marquee, just in front of the speakers.

08 Applying flat vinyls

If you have a full-height cabinet or a bartop and stand, you'll probably have a number of flat, front-facing areas to decorate – in our case, the front cupboard door of our stand, its base, and the front of its foot. Do these next to get your hand in.

The drill is the same for all of them: place the vinyl decal face-down on the floor, remove its backing, spray both it and the surface you're



applying it to, position your decal, and smooth it out with your applicator. Use a scalpel to trim off any overmatter. For the door, we applied the decal with the door in place – knob removed, starting at the top. We had to open the door to flatten and trim the vinyl in places.

09 Control panel decals

Most control panel decals wrap around the top and front of your panel. Buttons and joysticks should not be present during application. This is a relatively easy section to apply, but watch your position if there are decorative patterns designed to surround specific buttons or joysticks.

You may need to trim overmatter from the sides with a scalpel to get the decal to fold over the front face properly. Be careful when smoothing the vinyl on this fold, as it can be prone to both trapped air bubbles and damage from the join beneath.

10 Cabinet positioning

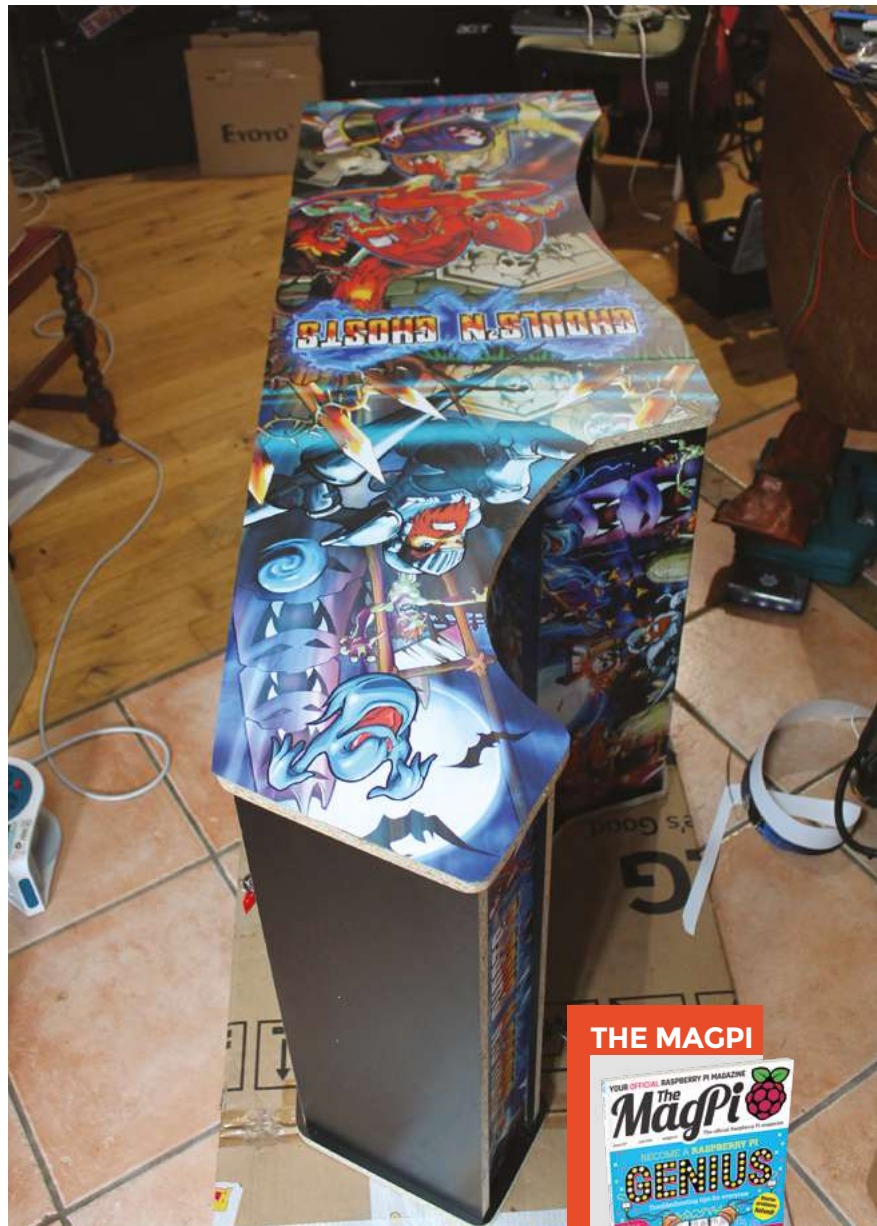
Side panels are the largest pieces of vinyl you'll be applying, but they're less intimidating than they seem. For a standalone bartop, one person can mount them in a vertical position with little fuss, as shown in Omniretro's video at magpi.cc/omniretrovinyl.

Full-height cabinets present more of a challenge due to their height and the size of the vinyl – a second person is useful here. You can apply long vinyls in an upright position, but we'd already attached rubber feet to our cabinet, so we used these to help pivot the cab down to lie on a sheet of cardboard on the floor.

11 Apply side panel vinyls

Lying flat and sprayed down as before, it's easy to line up the side-panel decal. Make sure everything's covered – with two people, it's easy to snap the decal back up if you make a mistake, then use a cloth and applicator to drive out excess moisture. Use a Stanley knife to trim the vinyl to size – its solid metal body makes it easy to follow the line of the cabinet's curves.

Go around again to remove any air bubbles and ideally leave the vinyl to dry for at least a couple of hours before pivoting the cabinet back up and lowering it to expose the opposite side. Repeat the process.

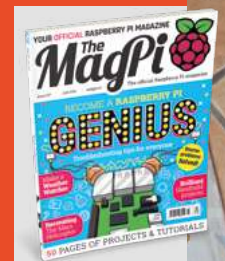


▲ You can leave some white-space overmatter on side panel decals before application, as they're easy to trim with a knife afterwards

“ Side panels are the largest pieces of vinyl you'll be applying ”

If your cabinet has separate stand and bartop parts, but uses a single sticker, there will be a slight ridge where these join. However, careful application (and a sympathetic vinyl design) makes this effectively invisible. Just be careful smoothing around it. →

THE MAGPI



This tutorial is from in The MagPi, the official Raspberry Pi magazine. Each issue includes a huge variety of projects, tutorials, tips and tricks to help you get the most out of your Raspberry Pi. Find out more at magpi.cc

Build an arcade machine: Decorate your cabinet

TUTORIAL



▲ Highly flexible, U- and T-moulding are used to give a clean finish to the cabinet edges



▲ Demonstrated here without glue, flex U-moulding backwards and use a finger or thumb to press it into place on a cabinet edge



Warning! Solvents

Always use solvents in a well-ventilated area and keep away from open flames.

magpi.cc/solvents

12 Moulding

We used U-moulding on our cabinet, with neoprene glue to hold it in place securely. First, measure and use scissors to cut two strips to go above and below the marquee – it's better to cut these a few millimetres too long and then trim than it is to have a gap.

Use a spatula to help apply neoprene glue along the edge you're working on, then use the tube's nozzle to apply glue to the inside of the U-moulding.

To lock U-moulding into place, bend it backwards to spread the U-shaped section, push that onto the edge you're applying it to, and then roll the moulding down along the edge, using a finger to push it into place.

When applying it to a long section, such as each side of your cabinet, start at the front underside – rubber feet help access here – apply glue to the cabinet edge and the first 50 cm of your roll of moulding, and have someone else feed it to you as you work up and around the cabinet. When you get to the bottom at the back, cut off your moulding with scissors.

T-moulding locks into a pre-cut groove along the edges of your cabinet, making it more secure, but it's still a good idea to apply glue to the flat surfaces for security. Either way, use a rubber mallet to gently tap down your moulding at the end.

You can use acetone to clean the glue off your hands and the moulding, but keep it away from the vinyl.



▲ After spraying the vinyl decal, and the acrylic, with our homemade fluid, we applied it and smoothed down with an applicator and cloth

Vinyl application fluid

You can buy commercial vinyl application fluid (magpi.cc/vinylfluid), widely used by car customisation enthusiasts to apply decals, but we filled a spray bottle with the following homemade formula:

- 66ml surgical spirit
- 132ml water
- 2 drops washing up liquid

You can use warm water with a drop of washing up liquid alone, but the surgical spirit reduces drying times, which means less waiting between different stages of application and decorating.

13 Finishing moves

Use a scalpel to cut out the vinyl above the button holes: locate a hole, pierce it with the blade, slice until you find the edge of the hole, and then follow the hole round to remove all the vinyl. Do this for all your joystick and button holes.

As described in *The MagPi* #105 (magpi.cc/105), screw your joysticks back into place from the inside. If you're going to put protective acrylic panels over your control panel, this is the time to do it – they're held on solely by the buttons.

However, because our cabinet is for home use, we've left the vinyl bare for a more comfortable and attractive finish. If your cabinet will see lots of play, acrylic will protect it and cut down on wear and tear. Whichever you choose, connect a DuPont cable to each button and pop them into place.

Follow the instructions from issue 106 to connect your buttons and peripherals to Raspberry Pi. ▣



Warning!
Sharp objects

Take care when using
knives and scalpels.

magpi.cc/handknives

FreeCAD, surfaces, and projection

Turn lines and sketches into solid objects



Jo Hinchliffe

🐦 @concreted0g

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!

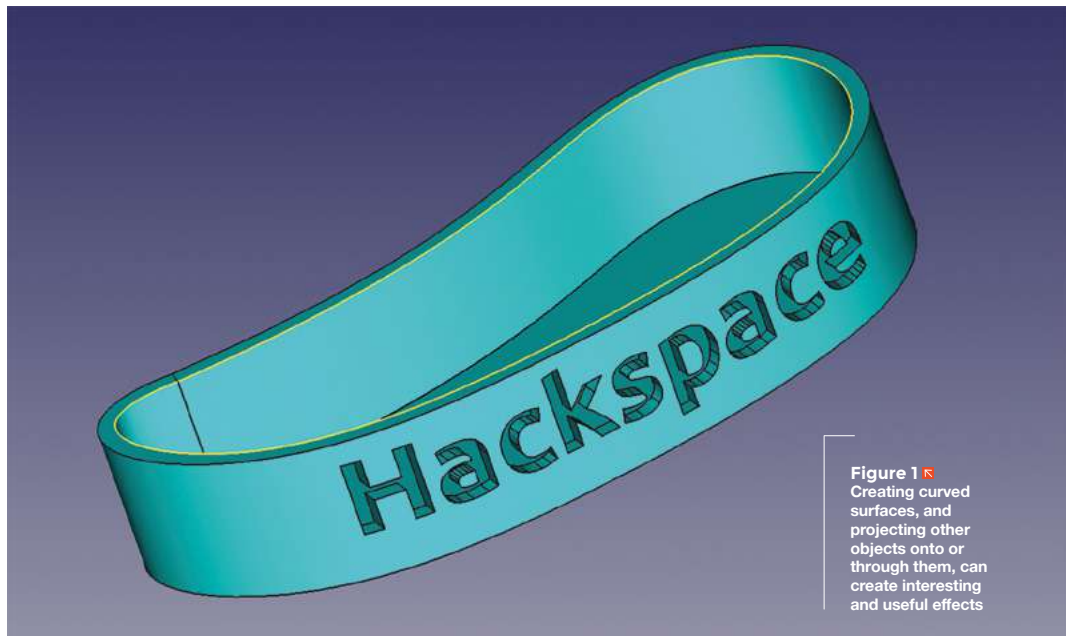


Figure 1 Creating curved surfaces, and projecting other objects onto or through them, can create interesting and useful effects



ften, we've begun with a sketch and then extruded it to create the first building block of our project or design. Similarly, here, we are

going to start with a sketch, but we are going to explore creating surfaces

from sketches or wires to build an object. To begin, let's go to the Sketcher Workbench and create a new sketch on the XY plane.

Select the 'B-spline by control points' option from the 'Create a B-spline in the sketch' drop-down in the drawing tools area, and draw a collection of B-splines around the origin point, making sure that the final B-spline ends at the first B-spline point (**Figure 2**). We aren't aiming for anything in particular, just an interesting curvy outline that has some inner and outer swooping curves. Play around moving the B-spline control points until you get a shape you like that

doesn't have too tight a set of curves (**Figure 3**). We don't need to constrain the B-spline that we have just created, so go ahead and close the sketch.

Moving back to the Part Workbench, select the sketch in the file tree, and copy and paste the sketch to create a second identical sketch that will appear as 'Sketch001' in the file tree. Select this second sketch in the file tree and, in the sketch dialog, increase the Z axis value in the Placement > Position drop-down to move the second sketch directly above the first. As a rough estimate, our curvy sketch is around 120mm at its longest point, and we moved our copied sketch upwards by around 25mm (**Figure 4**, overleaf).

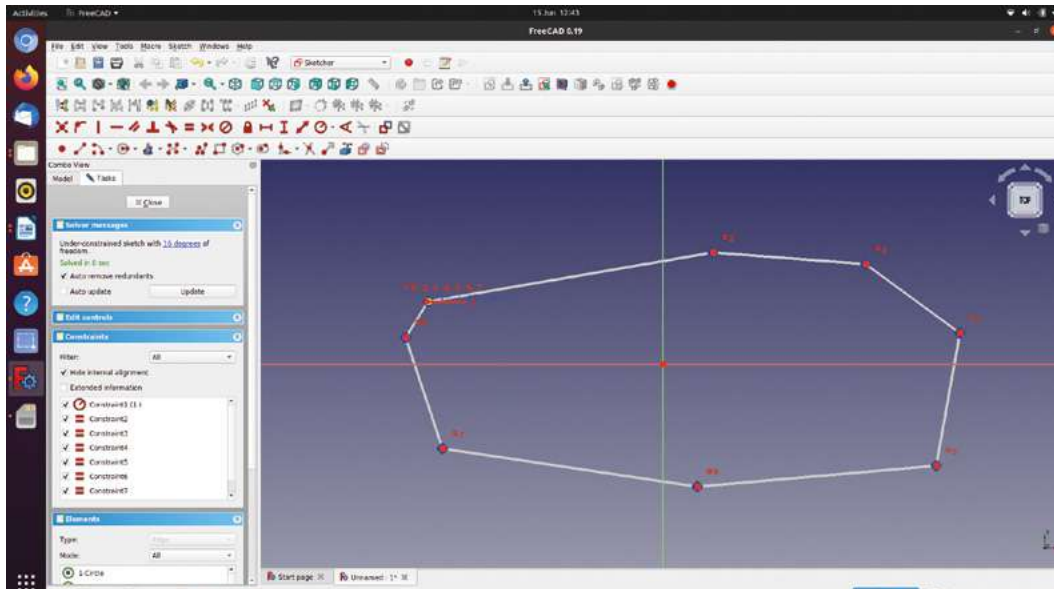
You should now see, in the preview view, two sketches perfectly aligned above one another. Select both the sketches in the file tree and let's create our first surface by clicking the 'Create a ruled surface' tool icon. Hopefully, you now see that a surface has

YOU'LL NEED

A computer and a copy of the 0.19 stable version of FreeCAD

QUICK TIP

We covered the basics of the Part Workbench and the Sketcher Workbench in the first two parts of this series, which started in issue 37.



been created between our two sketches, similar to our example in **Figure 5** overleaf.

You'll notice that, as it stands, our surface doesn't have any thickness and our object isn't a solid part with a top and a bottom. If we wanted to create a solid shape, of course, we would have used a single sketch and the pad or extrude tools. Instead, let's aim to make our curved wall into a sort of tray with a base, and turn our ruled surface into a wall with some thickness, but open at the top. First, let's make the ruled surface we created have some thickness but, just for fun, let's look

at the wrong way to do this using the extrude tool. Doing this incorrect experiment makes us think about the projection of axes through workpieces, which we will return to as a subject later in this article. Select your ruled surface and click the 'Extrude' tool; in the dialog, select the Y axis as a custom direction and set to extrude it a couple of mm. Apply the extrusion. Inspecting the results, you'll see that it works for some parts of the surface but not for others, as the extrusion is projected across the ruled surface in the Y axis. Applying an extrusion like this also leads to some parts of the extrusion being added inside the boundary of the ruled surface, but other parts extruded outside. Select the extrusion in the file tree and delete it, and toggle the ruled surface so that it is visible.

One of the best parts of using FreeCAD is the ability to apply multiple operations or tasks to a single piece of geometry. To add a base to our curvy wall, to make it into our curvy tray, we can use the drop-downs in the

file tree to go all the way back and select our original sketch. We can then click the Extrude tool and add -3mm to the Z axis custom direction and set the length option to 0. Clicking Apply, we should have added a base underneath the offset wall to complete our tray.

Previously in this series, when we have worked with solid objects, we have often created sketches

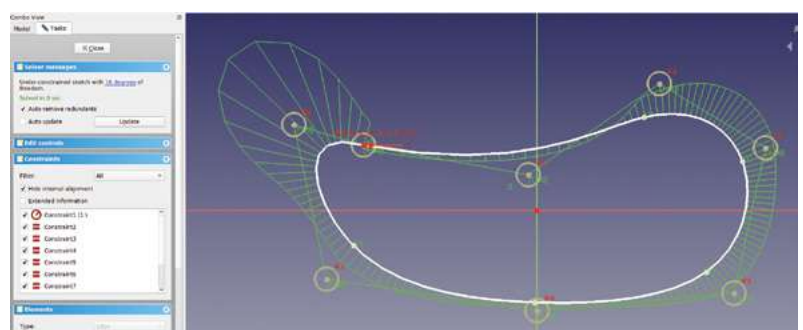
either attached directly to a flat face of an object or drawn on a plane, like XY or XZ, and then extruded or pocketed through an existing solid. With a curved object, such as the wall of our tray, drawing directly onto the surface is not really

possible. However, we might often want to add design aspects to a curved surface. One approach to this is to use 'projection'. Imagine placing your hand in front of a display projector and seeing the shadow of your hand on the projection surface on the wall or whiteboard. We can do this in FreeCAD, except the projector is referred to as the camera view and is simply the →

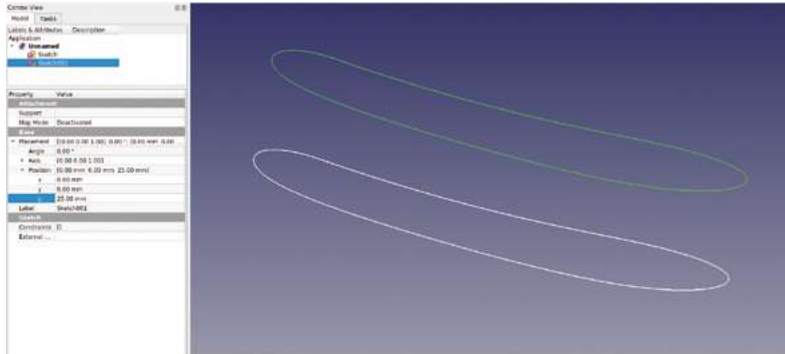
Doing this incorrect experiment makes us think about the projection of axes through workpieces

Figure 2 The rough arrangement of B-spline points before finishing the B-spline

Figure 3 Finishing our B-spline, we adjusted the control points to create a curved shape



TUTORIAL



current view in the preview window. The equivalent of your hand is the object you want to add to a surface, and our ruled surface is the object onto which we want to project. Whilst this might sound complex, it's actually pretty straightforward to do.

Whilst we have used the imported, and then extruded, dog logo for this first example of projection, you can use any small extruded part, perhaps a small extruded circle or square. First, we need to choose which surface area of our object we want to project the logo onto. You can project parts onto any surface, but obviously if the surface is very folded, the effect isn't going to work as well; therefore we chose a larger, smoother face. We then need to orient that chosen face to be pointing at us when viewed in the preview, remembering that the preview window is the projection viewpoint. With that established, we then need to move our object we want to project onto the curved surface into a position in line with the projected surface and the preview window. We can do this via the usual right-clicking the object in the file

tree, selecting Transform, and moving and rotating the object until we are happy with the position. It's not too critical how far away from the surface the object is, but it's a good idea to leave a reasonable gap so you can see the effects of the projection. In **Figure 7** overleaf, we have rotated the preview view to show how our projection was set up.

With your preview view and your projection object correctly set, click the 'Project edges, wires, or faces of one object onto a face of another object' tool. In the dialog you will see a long button that says 'Select projection surface' – click this button and then select the surface onto which we want to project. Next, click the 'Add face' button and select the face of the object you wish to project that is visible in the preview view. It may take a few seconds, but the selected face should turn a purple colour. You'll need to change the extrude height to a value that works for the shape of your projection surface. In our example, the extrusion height was 3mm – this gave enough height for the whole logo to be projected to the lowest parts and highest parts of the curve of the ruled surface. Experiment with values until you get a projection that works for you (**Figure 8**). Finally, click 'OK' and you should now see your object projected onto the curved surface of your tray part. Of course, now you can remove or toggle the visibility of the part you used to make the projection.

A common use for projection is to create 3D text effects. Let's first look at how to generate 3D text in FreeCAD, and then let's use some text and project it onto our curved surface again. Then we can combine projecting onto a curved surface with some Boolean part operations to create some interesting effects. For simplicity, let's just open a new project to look at

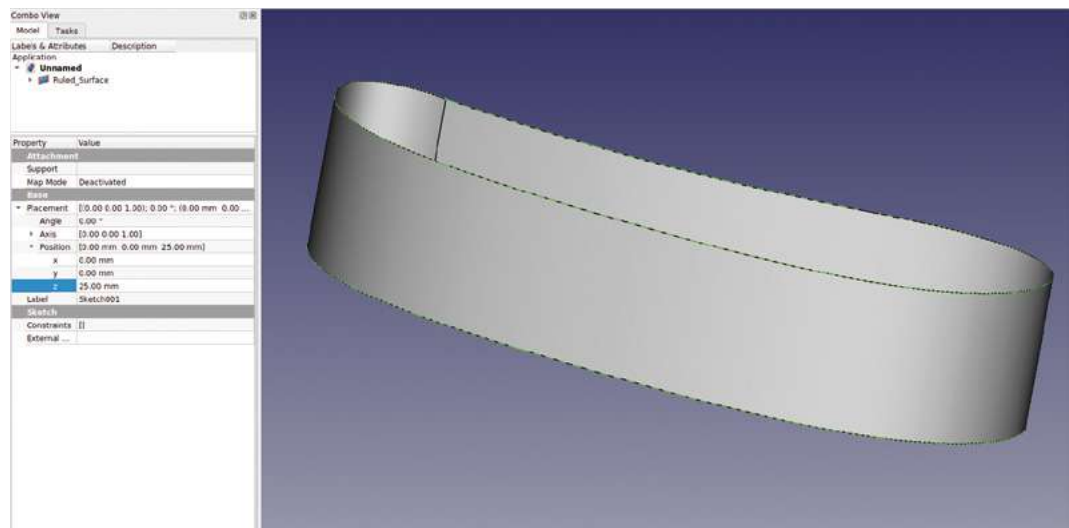


Figure 4 ♦
A copy of our B-spline sketch is created and raised above the original in the Z axis

Figure 5 ♦
Creating a ruled surface between two identical sketches

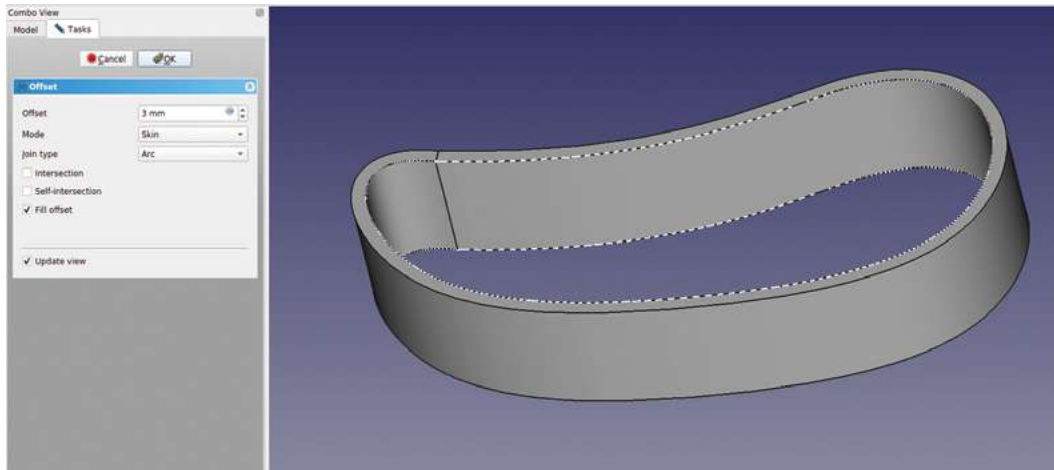


Figure 6 ♦ Creating a filled offset gives our ruled surface some thickness

the text tool we are going to use. In the new project, open the Draft Workbench and then click the 'Create a shape from a text string' tool icon, which looks like a yellow 'S'. In the dialog, you can see some positional co-ordinate input boxes which select the point at which our text will be created from. If you move your pointer over the grid in the preview window, you will see that these co-ordinates react to where your pointer is. For this example, let's leave them set at 0,0,0, which you can either type in or click the 'Reset point' button. You need to point the ShapeString tool

It's not too critical how far away from the surface the object is, but it's a good idea to leave a reasonable gap

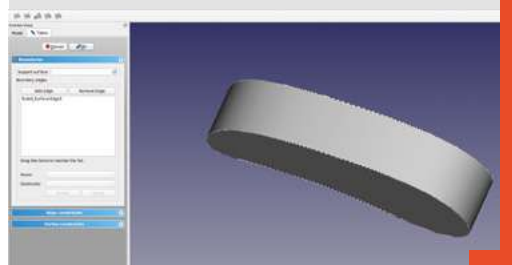
at your font installation folder. Clicking the ellipsis at the side of the 'Font file' input box should open a file browser. On Ubuntu flavours of Linux, you will need to right-click in the file browser and check the 'show hidden files' option, and font files can usually be found at **usr/share/fonts**. On Windows 10, the location is usually **C:\windows\fonts**. Navigate to your font folder and select a font. In the 'String' input box, type the text you want to create, and then set the height of the text, noting that the text height is not in the usual 'point' font units found in word processing packages, but rather, in millimetres. Clicking OK, you should now see the text string you input created in the draft preview window. Notice that it creates a wire/edge, but also creates a surface face. →

SURFACE WORKBENCH

Using the ruled surface tool on the Part Workbench is useful. In some ways, it is the first step towards thinking about 3D modelling in terms of edges and surfaces – rather than creating a 3D object via extrusion or padding, and then cutting and adding parts to and from it.

Surfaces in FreeCAD are given their own Workbench, and it's certainly worth exploring in its own right. The ruled surface tool we use in this article has some limitations – one main one being that, if you have a closed path like the base of our curvy tray, the ruled surface tool can't be used to create a surface filling the path. This is why, in the article, we create a small extrusion from the original sketch. If you are interested, create a sketch similar to the first B-spline sketch we created in the main tutorial, and then move to the Surface Workbench. Select the first tool icon, 'Creates a surface from a series of picked boundary edges', and in the dialog, click 'Add edge', and then select the sketch edge in the preview window. You should see a surface created filling the sketch area.

We hope to revisit the Surface Workbench and other surface tools in future articles, as it has a plethora of tools that enable complex operations in some ways similar to the lofting and sweeping operations we looked at in issue 40, but it's even more versatile.



QUICK TIP

You could also project onto a surface using the usual height method, and then use the Transform function to move the projection to a specific depth.

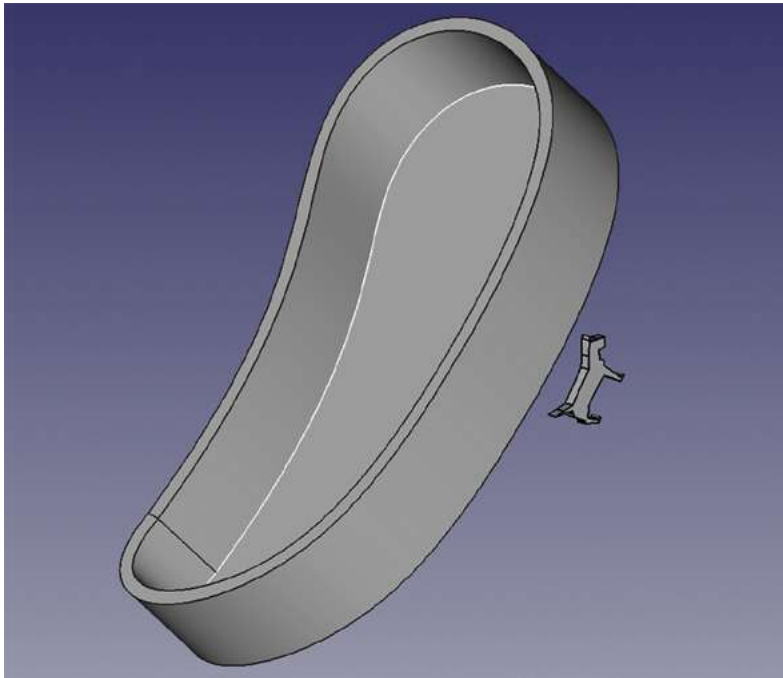
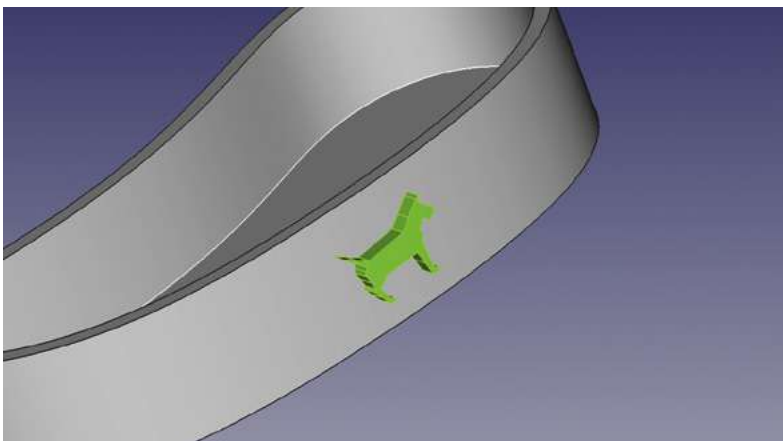


Figure 7 ♦
A rotated view showing the setup we created to project the logo part onto our curved tray wall

Figure 8 ♦
Our logo projected onto our curved part. It creates a useful and pleasing embossed effect



You should also see that you have a 'Shapestring001' object in the file tree (**Figure 9**). Now that you have a ShapeString object, you can move back to the Part Workbench and perform the usual sort of operations upon it. Whilst not totally necessary, if we extrude the ShapeString on the Part Workbench a little, say 3mm, it means that we can use the simple 'Transform' operation we used earlier to move the object into our projection position.

We can now use the text in the same way as we used our small logo or part in the earlier projection example. In **Figure 10**, you can see that we have again projected the text onto the curved side of our curvy tray example.

You may have noticed when we set up a projection, as well as a height input box, there is also a 'Depth' box. Of course, the depth option allows you to project

through the selected surface to a set depth. To see how this works, a useful experiment is to project into a part that is partially transparent. In a new project, create some example text using a ShapeString, and then return to the Part Workbench. Click the 'Create a cube solid' to create a cube. Resize the cube so that it is larger than your text ShapeString, and then right-



The surface tools and projection approaches can create very complex effects and designs



click and select Transform; move the cube item down underneath the ShapeString, ready for projection.

Click the Projection tool and select the surface of the cube item we just made. Then select the faces of the ShapeString in the usual way. Add 2mm of height, but add 5mm of depth to the projection, and then project onto the cube item. Select the cube item in the file tree, right-click, and select Appearance. In the appearance dialog, slide the 'Transparency' slider to the right to set the transparency to around 40%. Closing the Appearance dialog, you should now be able to see that the cube is partially transparent, and we can see the projection is inside the cube to a depth of 5mm. Finally, we can select the cube and then the projection object, and click the 'Make a cut of two shapes' tool to remove the projection from the cube object. This is a useful effect if you are trying to make a text-based sign object for 3D printing (**Figure 11**).

Figure 9 ♦
Making a ShapeString object is the basis of creating 3D text effects

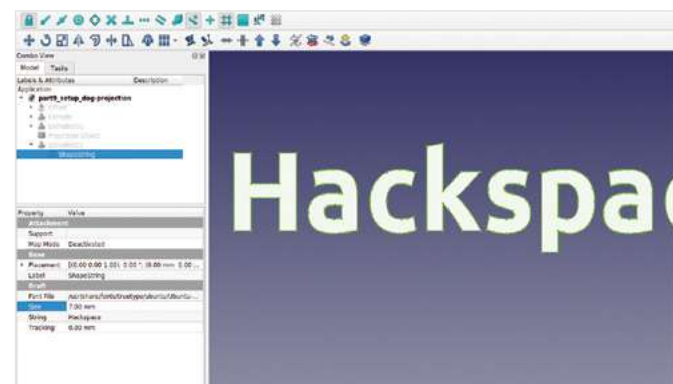




Figure 10 ♦ Combining our new skills, it's easy to project text onto curved surfaces



Figure 11 ♦ The depth setting in the projection dialog makes it easy to create cut-out projection effects

PUTTING IT ALL TOGETHER!

As a final experiment, we can put together all the techniques we've covered in this part of the series. We've returned to our slightly odd-shaped curved tray, and we have once again set it up with our example text ShapeString projected onto it. This time, we have kept our height at 10mm and set our depth to 10mm, which provided more than enough depth to fully project the text through the curved wall of the tray. You can see in above that it looks a little messy, but in **Figure 1** you can see the result when we have performed a cut operation to remove the projected object from the tray. It's an interesting fretwork-type effect that can make a very pleasing object, although for 3D printing we would have to model some struts to keep the floating parts of the lettering in position!

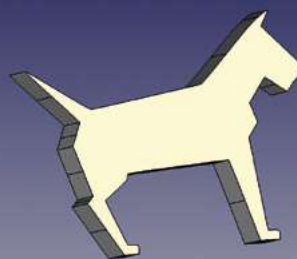
The surface tools and projection approaches can create very complex effects and designs, but we mentioned earlier that they have their limits. How, for example, could we create a pattern or text that flows completely around a cylinder or other curved shapes? Next month, we'll look at other tools and techniques to get us solving these issues, and more. ▢

VECTOR GRAPHICS

For our example of projecting a part onto a curved surface, we have used a small dog logo, which was imported as an SVG and then extruded into a part. The SVG file for the dog logo is a single path outline – experiments with more complex SVGs had varied results.

The ability to import an SVG is a really useful feature occasionally, and works well. We simply went to File > Import and navigated to our SVG. When selected, we get a dialog asking if we want to import as a 'Drawing' or 'SVG as Geometry'. We need to select the latter, and then the dog logo is imported and appears in the file tree as an object labelled 'Path'. On the Part Workbench, if we select the imported SVG and then click the Extrude tool, we can extrude in the usual way. We extruded the little dog logo to 2mm.

There are also ways to import an SVG and convert the SVG into a sketch so that we can constrain its contents, making it usable in other Workbenches. A useful video tutorial on this, by the author, is available here: hsmag.cc/ImportSVG.



Create a portable two-button game

Write a Python game and then play it on as many different hardware platforms as possible, including remotely over MQTT



Rob Miles

🐦 @robmiles

Rob Miles has been playing with hardware and software since almost before there was hardware and software. You can find out more about his so-called life at robmiles.com.



In this article, we are going to make a fun little game. Then we are going to discover how to make the game work on lots of devices and over MQTT. The game engine will be the same, but the low-level hardware interfaces are modified for each. Along the way, we'll find out a bit about software objects and class-based design.

This portable game project started some time ago at a Raspberry Pi Jam, which had the development theme of 'Two-Button Game'. The idea was to make a game controlled by just two buttons. The initial hardware used two push-buttons – a red button and a blue one – along with some NeoPixels and a Raspberry Pi Zero. The gameplay is simple: coloured lights are displayed on the pixels, and the player presses the red button if they see more red lights, or the blue button if they see more blue ones. The game fits nicely into a junction-box case.

Each time a player presses the correct button, the lights all flash green and the game displays another

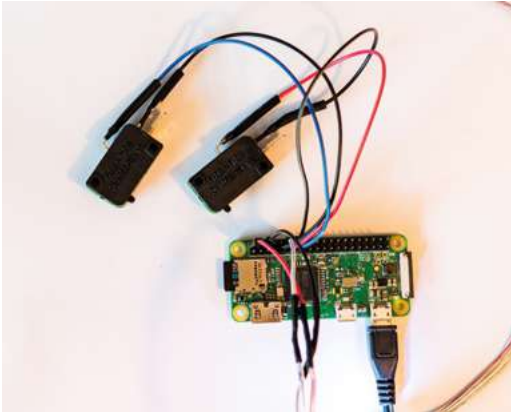
puzzle. When the player pushes the wrong button, the lights all flash red and the score is displayed; first yellow flashes to count the tens, and then purple flashes for units. The game gets progressively more difficult as the score increases. The colours start to move around the box, and when you get to 'ninja' level, the lights only show their colours for a brief instant. The game proved quite popular. When you see people fighting for a go on the game, you know that you are on to a winner. Let's have a look at how the software works.

RED VS BLUE

The first thing the game needs is a number of red lights and a number of blue lights for the player to count. The total number of lights must be less than six because this is the number of lights on the box, and the number of red lights must not equal the number of blue lights. This Python code is used to make the red and blue counts:

Above

The very first two-button game box. There are lights around the sides of the box, a red button on one end, and a blue button on the other. The power switch is a simple push-button. The blanking plugs for unused holes in the junction box make good diffusers for the NeoPixels inside.



Above

This is the wiring for version 1.0 of the game. The microswitches clip onto the bottom of the arcade buttons

```
import random
while True:
    red_count=random.randrange(1,6)
    blue_count=random.randrange(1,6)
    if (blue_count+ red_count) > 6:
        continue
    if blue_count==red_count:
        continue
    break
```

The code uses the random library to generate numbers in the range 1 to 5. This means that there will always be at least one red and one blue light. Note that the upper limit of `randrange` is exclusive, meaning that the value 6 is never produced. A `while` loop repeatedly picks random values for `red_count` and `blue_count` and checks that they can be used (less than six total lights and red and blue different). A `continue` statement is used to restart the loop if invalid values are detected, and a `break` breaks out of the loop if all the tests are passed. The loop might look a little inefficient in that if it is 'unlucky', it will have to make multiple attempts. However, testing indicates that the loop very rarely goes round more than three times.

PUZZLING LIGHTS

The red and blue count values are now used to build a `puzzle` list holding the colour values for display on the lights. A colour is made up of three intensity values in the range 0 to 255. The first value is for red, the second green, and the third blue. The game uses eight colours, which are defined at the start:

```
red=(255,0,0);green=(0,255,0);blue=(0,0,255)
cyan=(0,255,255);magenta=(255,0,255);
yellow=(255,255,0)
white=(255,255,255);black=(0,0,0)
```

The game builds the puzzle by putting red and blue colour values into successive locations of the puzzle

list. The variable `pos` is used to specify the position of each colour and is moved along after each colour has been added.

```
pos=0
for i in range(red_count):
    puzzle[pos]=red
    pos += 1
for i in range(blue_count):
    puzzle[pos]=blue
    pos += 1
```

Once we have put the red and blue colours into the puzzle, the next step is to add some colours for the rest of the lights:

```
for i in range(pos,6):
    puzzle[i]= random.choice(fillers)
```

Remember that the variable `pos` holds the position in the puzzle that was reached when we were filling it with red and blue values. The `for` loop, above, will continue from `pos` up to the end of the puzzle. The `choice` function in `random` picks a random item from a collection. The `fillers` variable holds a collection of colours used to fill up the display:

```
fillers=(green,cyan,magenta,yellow,white)
```

The final thing that the game does is shuffle the puzzle ready for display: →

```
1 while True:
2
3     score=0
4     alive=True
5
6     while alive:
7         get_puzzle()
8         shuffle_puzzle()
9         show_puzzle()
10        start_time=time.time()
11        while True:
12            red_button_down = not red_button.value
13            blue_button_down = not blue_button.value
14
15            if red_button_down and blue_button_down:
16                alive=False
17                break
18
19            if red_button_down:
20                if red_count > blue_count:
21                    score=score+1
22                else:
23                    alive=False
24                    break
25
26            if blue_button_down:
27                if blue_count > red_count:
28                    score=score+1
29                else:
30                    alive=False
31                    break
32
33            if (time.time() - start_time) > response_time:
34                alive=False
35                break
36
37            if alive:
38                flash(green,0.5)
39                wait_for_buttons_released()
40
41            flash(red,0.5)
42            display_score(score)
```

YOU'LL NEED

- ◆ A Raspberry Pi Zero, Raspberry Pi Pico, or ESP32 device
- ◆ NeoPixels and push-buttons
- ◆ The Pico RGB Keypad Base from Pimoroni
- ◆ A PC and some Connected Little Boxes with buttons and lights

Left

This is the overall structure of the game loop. Some behaviours have been put into functions to make the loop smaller. The `flash` function flashes all the pixels with the specified colour for the specified time. The `display_score` function flashes the pixels to show the score

Creating a portable two-button game

TUTORIAL

```
File Edit Tabs Help
GNU nano 3.2 /etc/rc.local Modified
#
# By default this script does nothing.
#
# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi
sudo python3 /home/pi/Desktop/Portable2Button.py &
exit 0
AG Get Help  AR Write Out  AW Where Is  AK Cut Text  AJ Justify  AC Cur Pos
AM Exit      AR Read File  AN Replace  AU Uncut Text  AT To Spell  AA Go To Line
```

Above

The program is in the file 'Portable2Button.py', which is stored on the desktop. The statement above exit is the one that will run the game. Note that the command to run the game is followed by the '&' character, which will allow the rc.local file to complete running when the machine starts. When the Raspberry Pi boots it will now automatically run the game. You can make a Raspberry Pi start more quickly by installing a 'headless' version of the OS. Take a look at Raspberry Pi OS Lite

```
for shuffle in range(0,6):
    i = random.randrange(0,6)
    j = random.randrange(0,6)
    puzzle[i], puzzle[j] = puzzle[j],puzzle[i]
```

The shuffle code repeatedly picks random locations in **puzzle** and then swaps their contents over. The last statement in the code above does something that may be unique to Python; it lets you swap the contents of two variables with a single statement.

PIXEL PUSHING

The next thing to do is display the colours in **puzzle** on the NeoPixels. The first version of the game was written in Python on a Raspberry Pi and used the Adafruit NeoPixel drivers to drive the LEDs. This driver implements the pixels as a collection, and the game changes the colours of pixels by putting colour values into this collection.

```
import neopixel
import board
import digitalio
pixels = neopixel.NeoPixel(board.D18, 6)
pixels[0]=red
```

The first statement imports the **neopixel**, **board**, and **digitalio** libraries for use by the pixels and the buttons. The final two statements create an object called **pixels** that contains six pixels and is connected to pin D18 on the Raspberry Pi and then display the red colour on the first pixel (remember that, in Python, storage is indexed starting at 0). The loop below copies the contents of **puzzle** onto the pixels.

```
pos=0
for colour in puzzle:
    pixels[pos]=colour
    pos = pos + 1
```

BUTTON PUSHING

Now that we have the lights displayed, it's time for some user input. The player has four seconds to press either the red or blue button. The game uses the Raspberry Pi **digitalio** and **board** libraries to read the states of the two buttons.

```
red_gpio=board.D22
blue_gpio=board.D27

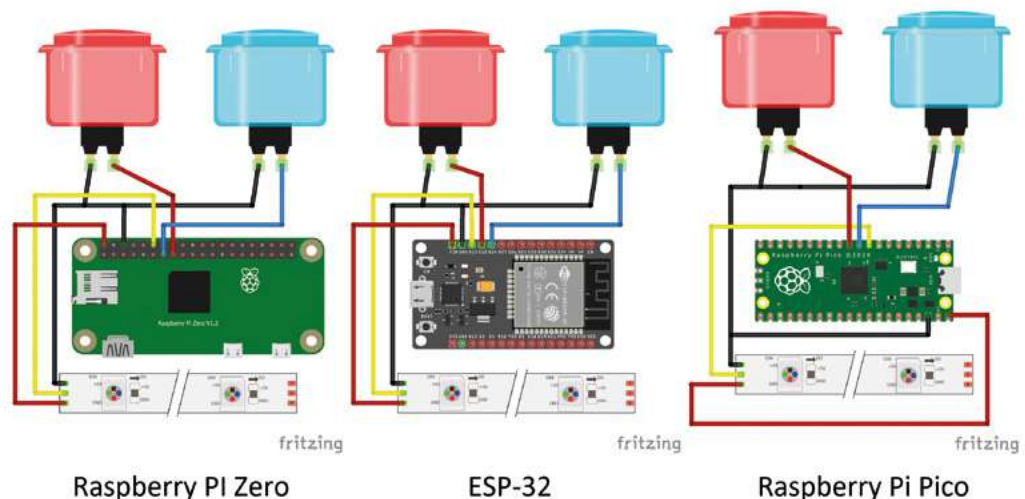
red_button = digitalio.DigitalInOut(red_gpio)
red_button.direction = digitalio.Direction.INPUT
red_button.pull = digitalio.Pull.UP

blue_button = digitalio.DigitalInOut(blue_gpio)
blue_button.direction = digitalio.Direction.INPUT
blue_button.pull = digitalio.Pull.UP
```

The buttons are implemented by variables called **red_button** and **blue_button**. These are **DigitalInOut** values that are configured as inputs and given pull-up resistors in the code above. The pull-up resistors

Right

These are the circuits that we can use for each of the devices. If the number of LEDs is small, they can be powered from the power output pin of each device



setting asks the Raspberry Pi hardware to lift the input pins into the high state. Then the push-buttons just have to connect a pin to ground when the button is pressed. This means that the input will be inverted (the signal will be low when the button is pressed), but the code can invert this when the button is read:

```
red_button_down = not red_button.value
blue_button_down = not blue_button.value
```

The two statements above set the variables `red_button_down` and `blue_button_down` for use in the game. The signal state is read from the `value` property. Now the game can implement a test to see if the player has pressed the correct button:

```
if red_button_down:
    if red_count > blue_count:
        score=score+1
    else:
        alive=False
        break
```

The code above is for the red button. If it has been pressed and the red count is greater than the blue count, the player has got it right (yay!). The `score` is increased by one. If the player has got it wrong (boo!), a flag called `alive` is set to `False` to indicate that this game is over.

GAME STRUCTURE

The game uses a number of nested `while` loops. The outer loop runs forever. This is the loop that plays multiple games. The inner loop runs while the player is alive. This contains a third loop that deals with button presses.

PORTABLE HARDWARE

The game works well, but it does use an entire Raspberry Pi Zero, along with an SD card. There are much cheaper devices that also run Python, for example, the ESP32 and Raspberry Pi Pico, so it would be nice to be able to use one of them. You can install MicroPython on an ESP32 device by following the instructions at hsmag.cc/MP_ESP32. A guide to running Python on a Raspberry Pi Pico can be found at hsmag.cc/Pico_GetStarted.

Unfortunately, although the wiring for them is very similar, each of these devices has its own way of talking to hardware from a Python program. Rather than having to write different versions of the game for each bit of hardware, we can instead use a bit of programming cleverness to separate the game engine (the bit that manages the gameplay) from the hardware platform (the bit that displays pixels

```
1 class RPi(object):
2
3     def __init__(self):
4         import digitalio
5         import board
6         import neopixel
7         neo_gpio=board.D18
8         red_gpio=board.D22
9         blue_gpio=board.D27
10        self.no_of_leds = 6
11
12        self.pixels = neopixel.NeoPixel(neo_gpio, self.no_of_leds)
13
14        self.red_button = digitalio.DigitalInOut(red_gpio)
15        self.red_button.direction = digitalio.Direction.INPUT
16        self.red_button.pull = digitalio.Pull.UP
17
18        self.blue_button = digitalio.DigitalInOut(blue_gpio)
19        self.blue_button.direction = digitalio.Direction.INPUT
20        self.blue_button.pull = digitalio.Pull.UP
21
22    def pixels_show(self):
23        pass
24
25    def pixel_set(self,i, col):
26        self.pixels[i]=col
27
28    def red_down(self):
29        return not self.red_button.value
30
31    def blue_down(self):
32        return not self.blue_button.value
```

and reads buttons). We can do this by creating software objects.

THE OBJECTS OF OUR DESIRE

A software object is a bit like a person, in that you can tell it things and ask it questions. You do this by calling methods contained in the object. Each method contains some Python code that runs when the method is called. A program can pass values into a method (for example, the number and colour of the pixel to be lit), and a method can return a value (for example, the state of the blue button).

We are going to create an object which acts as a 'platform' for our game code. If we consider what our game does with the hardware, we can boil it down to the following behaviours:

- Set a new colour for a pixel
- Update all the pixels to their new colour
- Test if the red button is down
- Test if the blue button is down

We are going to create a design for an object which can perform these functions. Then we are going to make several versions of this object, one for each different hardware platform that we want to use. Then we just tell the game to use a particular platform object, and the game will work on that →

Above

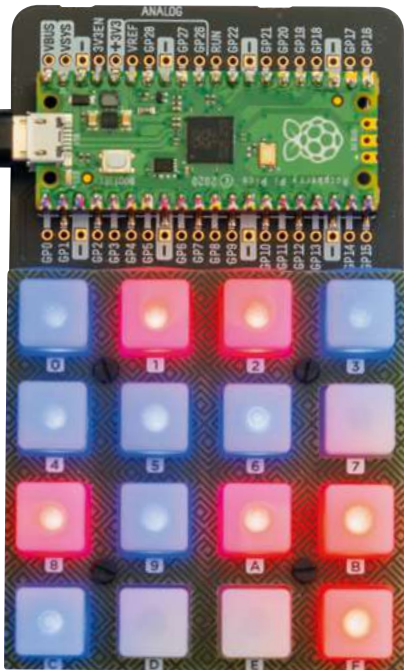
This is the hardware platform class for the Raspberry Pi. Each of the behaviours is implemented by a method. The `pixel_set` and `pixels_show` methods just perform an action, whereas the `red_down` and `blue_down` methods return `True` if the corresponding button is pressed down. Note that the `pixels_show` method is empty for this device because the Raspberry Pi NeoPixel driver always updates the pixels when one is changed. The `__init__` method is a special method in a Python class that is given control when a new class object is created. The class uses the `__init__` method to set itself up. When we want to make a hardware platform for the ESP32 or Raspberry Pi Pico, we create an object that contains methods and make sure that they behave as the game engine expects

TUTORIAL

Right ♦
The coloured lights are displayed around each button. At this point in the game it might be a good idea to press the red button



Below ♦
This version uses the top twelve keys as pixels and two buttons on the bottom row (locations C and F) to make the red or blue selections



platform. You might be wondering why setting a new pixel colour and updating the pixels have been made separate actions. This is to allow the platform to buffer several colour changes, which can then be performed by updating all the LEDs at once. Implementing a change to one pixel by rewriting all of them might make them appear to flicker as each individual pixel was changed in turn.

RUNNING THE GAME ENGINE

The game engine is another object which is called **two_button_game**. This object contains a method called **play** which is called to start the game. The **play** method contains the game loop that we saw earlier.

```
platform = RPi ()
game = two_button_game(platform)
game.play()
```

The code above shows how a game is started. The first statement creates a variable called **platform**, which refers to a hardware object containing the behaviours for a Raspberry Pi device. The next statement creates a variable called **game** which refers to a **two_button_game** object. When the program creates the **two_button_game** object, it passes it the platform that it is going to use. The final statement calls the method **play** on the game to start the game running. The game calls methods on the platform to interact with the player.

BRANCHING OUT

There are lots of advantages in working this way. We can also create new games that use buttons and lights, and they will run on any hardware platform. We can add new platforms without caring how the game works. The Pico RGB Keypad Base, from Pimoroni, provides an enticing mix of buttons and coloured lights, along with a set of driver routines to set colours on the buttons and detect when they are pressed. With an appropriate hardware driver, the game can be made to work in that device.

// The Pico RGB Keypad Base from Pimoroni provides an enticing mix of buttons and coloured lights //

MOVING TO MQTT

We can even use MQTT (Message Queue Telemetry Transport) messages to implement a distributed version of the two-button game. The game engine will run on a server (a PC or Raspberry Pi). The server will receive messages about button state and send commands to set the colours of the pixels in the remote devices. MQTT is a way of connecting embedded devices. A central broker accepts incoming messages and re-publishes them to devices that have registered an interest in a particular topic. If you've not investigated MQTT before, you really should take a look. You can run your own broker for local devices on your private

network, or you can use a remote broker to interact with devices anywhere on the internet.

The buttons contain ESP8266 devices running the Connected Little Boxes software. This uses MQTT to send and receive messages. You can find out more in issue 41 of HackSpace magazine. Each button is running the following Connected Little Boxes command:

```
{ "process": "console", "command": "reportjson",
  "attr": "button", "sensor": "button", "trigger": "changed",
  "to": "gamehost" }
```

This tells the box to send a JSON-formatted message to the topic 'gamehost' each time the button sensor changes state. This is the message that is sent to 'gamehost' when the button is pressed:

```
{ "process": "console", "command": "reportjson",
  "text": "down", "attr": "button", "from": "blue" }
```

The message contains the state of the button (the text value) and the device name of the sender (in this case, 'blue'). The MQTT platform for the game uses the Paho MQTT client to listen to the 'gamehost' topic. It connects a function to incoming MQTT messages, which sets the state of the button whenever it receives an update:

```
def on_message(self, client, userdata, message):
    message_text = str(message.payload.decode("utf-8"))
    message_obj = json.loads(message_text)
    if message_obj["from"] == "blue":
        if message_obj["text"] == "down":
            self.blue_down_flag = True
        else:
            self.blue_down_flag = False
```

The function decodes the JSON received and checks to see if the message is from the blue button. If it is, the function checks the text attribute to see if it is 'down' and then sets a flag to indicate the button state.

```
def blue_down(self):
    return self.blue_down_flag
```

Above is the `blue_down` method from the MQTT platform – it just returns the contents of the flag. The server sends Connected Little Boxes commands to set the colours of the pixels in the boxes and display the user interface.

USING THE GAME

You can find the code at: hsmag.cc/TwoButtonGame.

The game is supplied as a single file of Python code, which makes it easy to install on embedded devices.

You select the device to be used by setting the `version` variable at the top of the file:

```
# Uncomment the version that you want to run.
version="RPI"
#version="ESP32"
#version="PICO"
#version="PICO-RGB"
#version="CLB"
```

The settings above configure the game to work on a Raspberry Pi device. Note that if you select the CLB version you will also have to configure the MQTT settings in the code to connect to the MQTT server. You will also have to configure two devices to send button messages. Full details of how to do this are here: hsmag.cc/CLB_Projects ▣

MAKING A RASPBERRY PI INTO AN EMBEDDED DEVICE

If you want to make a Raspberry Pi into an embedded device, you'll need to install the hardware drivers so that Python can talk to the hardware. You'll also have to make the game program run automatically when the Raspberry Pi starts.

```
sudo pip3 install rpi_ws281x adafruit-circuitpython-neopixel
```

Use this command in the Raspberry Pi command console to install the required drivers. If you want to use the Thonny development environment to work on Python programs that interact with hardware, you will have to run Thonny with superuser privileges. Don't start it from the desktop, instead, run it from the command prompt using the `sudo` command:

```
sudo Thonny
```

The final thing you need to do is make your game program run when the Raspberry Pi starts up. You can add a line to the `rc.local` system file. This file contains commands that are performed when the Raspberry Pi starts. Edit this file with the nano editor using this command:

```
sudo nano /etc/rc.local
```

```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
GNU nano 3.2 /etc/rc.local Modified
#
# By default this script does nothing.
# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi
sudo python3 /home/pi/Desktop/Portable2Button.py &
exit 0
AG Get Help  AG Write Out  AN Where Is  AK Cut Text  AJ Justify  AC Cur Pos
AX Exit      AR Read File  AN Replace  AU Uncut Text AT To Spell  AL Go To Li
```

THE OFFICIAL Raspberry Pi Beginner's Guide

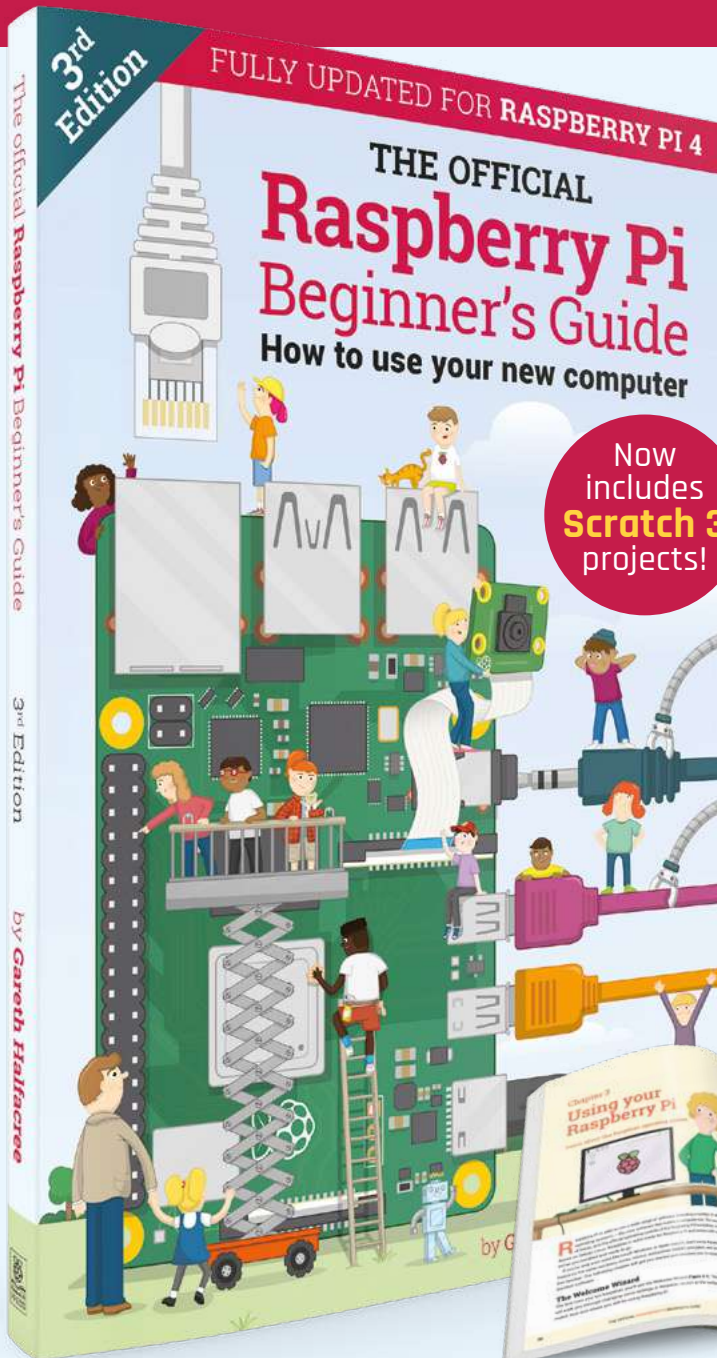
**The only guide you
need to get started
with Raspberry Pi**

Inside:

- Learn how to set up your Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch 3 and Python languages
- Create amazing projects by connecting electronic components to Raspberry Pi's GPIO pins

Plus much, much more!

**£10 with FREE
worldwide delivery**



Buy online: magpi.cc/BGbook

Wireframe

Join us as we lift the lid
on video games



Visit wfmag.cc to learn more

Program the Nibble handheld console

Build a simple game for the EPS8266-based console



Ben Everard

@ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

Below

You can add more images to make a complete emoji game

W

e've been playing with the CircuitMess Nibble console as part of the review process.

While it's a fun bit of kit, there's not much information to help new people get up

to speed with the Arduino library. In this tutorial, we'll look at getting started with CircuitOS, and the basics you need to know if you want to build your own game. You might need to do a bit of digging around, so this is more suitable for people with a bit of programming experience. If you're a new programmer, you might find it easier to get started with the CircuitBlocks programming environment (hsmag.cc/CMCircuitBlocks).

First of all, you'll need to install the Arduino IDE (if you don't already have it) and install the CircuitMess Nibble board definition. See here for more details: hsmag.cc/CMArduino.

Nibble is a little unusual in that when you select the board, you automatically have access to the libraries to run the hardware, so you don't need to install these separately. Let's take a look at how to use these

libraries to interact with the different bits of hardware on Nibble.

First, we'll take a look at getting output from Nibble. Here's a basic sketch that initialises the hardware and sends a few outputs:

```
#include <Nibble.h>
Display* display;
Sprite* sprite;

void setup() {
  Nibble.begin();
  display = Nibble.getDisplay();

  Serial.begin(115200);
  sprite = display->getBaseSprite();
  sprite->clear(TFT_BLACK);
  display->commit();

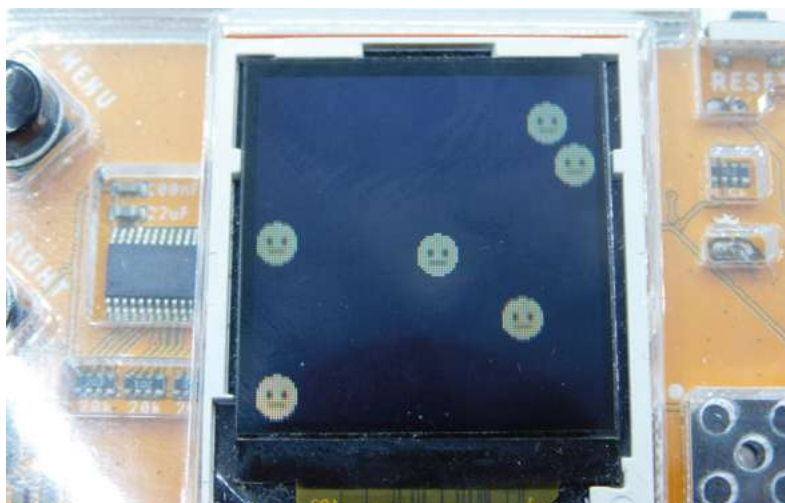
  Piezo.begin(BUZZ_PIN);

  sprite->setTextColor(TFT_WHITE);
  sprite->setTextFont(1);
  sprite->setTextSize(1);
  sprite->printCenter("hello world");
  display->commit();

  Piezo.tone(440, 500);
}

void loop() {
  Serial.print("hello from nibble\n");
  delay(1000);
}
```

You might notice that we're using some things that we haven't defined, such as **Nibble** and **Piezo**. These are defined in the header files. Unfortunately, there's no documentation on exactly what is available to us and how to use it. If you want to delve into what you've got to work with, the best place to look is in the source code. The two main parts are the Nibble



repository ([hsmag.cc/CMNibble](https://github.com/hsmag/cc/CMNibble)) and the CircuitOS repository ([hsmag.cc/CMCircuitOS](https://github.com/hsmag/cc/CMCircuitOS)). They're not particularly huge, so it's usually straightforward enough to find out what you need to know by poking through the header files.

By including **Nibble.h**, we get all the hardware libraries we need, so we don't need to worry about including them. The **Nibble.begin()** call sets all the hardware up, so we don't have to do it manually. However, we do want objects that we can use to control the screen and the main sprite. We create pointer variables to hold them, and then call the appropriate **get** methods to get the correct pointers.

As you can see, the main ways of getting data out of Nibble are the display, the buzzer, and the serial.

Let's now take a look at how to get data in. The main method for this is the buttons. There are six in total – four directions and two actions. They're all treated in the same way.

They're a little different from how you might have used buttons previously with Arduino because they're managed by the CircuitOS input library. This means that we don't have to worry about checking pin levels or debouncing. Instead, we can set **callback** functions to run when a particular event happens. Let's take a look at how this works:

```

#include <CircuitOS.h>
#include <Nibble.h>

Display* display;
input = Input::getInstance();
Sprite* sprite;

const int icon_diameter = 16;

int position_x = 60;
int position_y = 60;

bool up_held = false;
bool down_held = false;
bool left_held = false;
bool right_held = false;

bool beep = false;
bool boop = false;

const unsigned short face[0x100] = {...}

void up_press_callback() { up_held=true; }
void up_rel_callback() { up_held=false; }
void down_press_callback() { down_held=true; }
void down_rel_callback() { down_held=false; }

```

```

void left_press_callback() { left_held=true; }
void left_rel_callback() { left_held=false; }
void right_press_callback() { right_held=true; }
void right_rel_callback() { right_held=false; }

void a_held_callback() { Piezo.tone(440, 500); }

void b_held_callback() { Piezo.tone(880, 500); }

void setup() {
  Nibble.begin();
  display = Nibble.getDisplay();
  Piezo.begin(BUZZ_PIN);

  //button presses
  input->setBtnPressCallback(BTN_LEFT, left_press_
callback);
  input->setBtnReleaseCallback(BTN_LEFT, left_rel_
callback);
  input->setBtnPressCallback(BTN_RIGHT, right_
press_callback);
  input->setBtnReleaseCallback(BTN_RIGHT, right_
rel_callback);
  input->setBtnPressCallback(BTN_DOWN, down_press_
callback);
  input->setBtnReleaseCallback(BTN_DOWN, down_rel_
callback);
  input->setBtnPressCallback(BTN_UP, up_press_ →

```

DRAWING IMAGES

There are two main ways of drawing images on a sprite: **drawIcon** and **drawMonochromeIcon**. They work in very similar ways. They both take five mandatory parameters: a data array, the X position, the Y position, the width, and the height. Then there are two optional parameters: scale and colour.

In the case of **drawIcon**, the data array should have the data type unsigned short, and each element in it stores data in 565 format, which means that five bits are for red, six for green, and then five for blue. Find details on how to convert images to this format at [hsmag.cc/CMConvert](https://github.com/hsmag/cc/CMConvert).

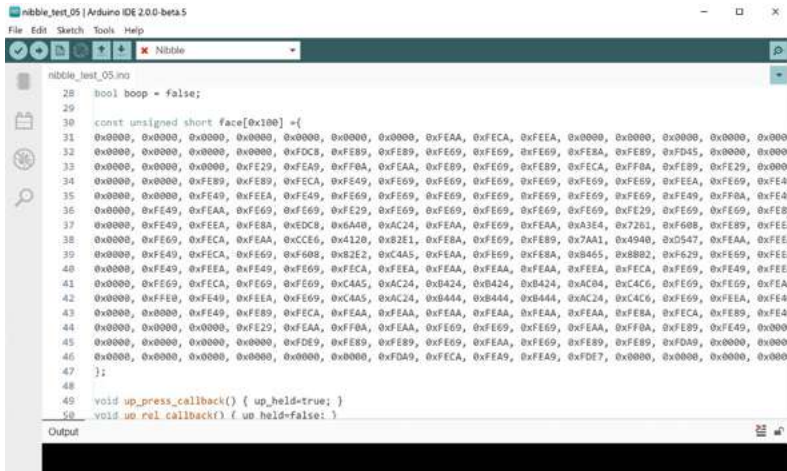
This means that **drawIcon()** has full-colour images. The final colour parameter is used to create a masking colour. Any items in the image that are this colour won't be drawn.


The **drawMonochrome()** method takes a data array that has the Boolean type, and each item is simply true or false depending on whether or not you want that pixel coloured. The colour parameter is the colour you want the dots drawn.

Take a look at the **nibble_test02** and **nibble_test03** sketches in the folder for examples of how these work.

Program the Nibble handheld console

TUTORIAL



Above  Icons are stored as unsigned short numbers. Each 16 bits of data stores five bits of red, six bits of green, and five bits of blue

```
callback);
input->setBtnReleaseCallback(BTN_UP, up_rel_
callback);

//beep boop
input->setButtonHeldCallback(BTN_A, 50, a_held_
callback);
input->setButtonHeldCallback(BTN_B, 50, b_held_
callback);

Serial.begin(115200);

sprite = display->getBaseSprite();
sprite->clear(TFT_BLACK);
display->commit();
}

void loop() {

  Input::getInstance()->loop(0);

  if (up_held) position_y--;
  if (down_held) position_y++;
  if (left_held) position_x--;
  if (right_held) position_x++;

  //note -- signed int here is important!!
  if (position_x > (signed int)display->getWidth()
- icon_diameter) {
    position_x = display->getWidth()- icon_
diameter;
  }

  if (position_x == -1) { position_x = 0; }
  if (position_y > (signed int) display->getHeight()
- icon_diameter) {
    position_y = display->getHeight() - icon_
diameter;
  }
  if (position_y == -1) { position_y = 0; }
```

```
sprite->clear(TFT_BLACK);
sprite->drawIcon(face, position_x, position_y,
16, 16, 1);
display->commit();

delay(10);
}
```

We haven't included the full data for the face icon because it just fills up space – take a look at the repository for the full code.

In each case, we have a function with no parameters that does the actions we want to happen when each button is pressed and released. In most cases, we're using this to set a global variable to let us know whether or not the button is currently pressed. However, you can do almost anything with it.

In the main loop, we first have to run the **Input** loop method which reads the buttons and calls functions as necessary. We then use the global variables for the button states to update the position of our icon.

One potential trap is that **display->getWidth()** returns an unsigned number. Comparing that to a signed number (we've declared **position_x** to be a signed integer) can create unexpected results if **position_x** is negative. We won't go all the way into C's type system here, but we got around the problem by casting the result of **display->getWidth()** to a signed number before doing the comparison.

When animating a sprite, you first need to blank it using the **clear** function, then you can draw on top of this, and finally run **commit** to push your changes to the screen.

MAKING A GAME

So far we've looked at inputs and outputs, but how do we turn this into a game? Let's create a simple game where one face that you control has to avoid falling faces (yes, the graphics are very rudimentary – we wanted to focus on the gameplay, so we'll leave making it look better as an exercise for the reader).

The code for this is mostly the same as the previous example, so we won't include it all here. Here are the new bits:

```
#include <CircuitOS.h>
#include <Nibble.h>

#define MAX_ROCKS 10
#define INIT_ROCKS 1
#define ROCK_INCREASE_SPEED 100

// omitted
```



```
void setup() {
  Nibble.begin();
  display = Nibble.getDisplay();
  Piezo.begin(BUZZ_PIN);
  // omitted

  for(int i=0; i<num_rocks; i++) {
    rocks_x[i] = random(0,display->getWidth());
    rocks_y[i] = 0;
  }
}

void loop() {

  loop_counter++;
  if(loop_counter > 100) {
    loop_counter = 0;
    if (num_rocks < MAX_ROCKS) {
      rocks_x[num_rocks] = random(0,display-
>getWidth());
      rocks_y[num_rocks] = 0;
      num_rocks++;
    }
  }

  Input::getInstance()->loop(0);

  // omitted

  sprite->clear(TFT_BLACK);
  sprite->drawIcon(face, position_x, position_y,
16, 16, 1);

  for(int i=0; i<num_rocks; i++) {
    rocks_y[i]++;
    if (rocks_y[i] > display->getHeight()) {
      rocks_x[i] = random(0,display->getWidth());
      rocks_y[i] = 0;
    }
    sprite->drawIcon(face, rocks_x[i], rocks_y[i],
16, 16, 1);
  }
  display->commit();

  //collision detection
  for(int i=0; i<num_rocks; i++) {
    int delta_x = position_x - rocks_x[i];

    int delta_y = position_y - rocks_y[i];
    if (delta_x*delta_x + delta_y*delta_y < 64 ) {
// 64 == r^2
      sprite->clear(TFT_BLACK);
      sprite->setTextColor(TFT_WHITE);
```




The game continues until the collision detection notices that your face collides with a falling face



```
      sprite->setTextFont(1);
      sprite->setTextSize(1);
      sprite->printCenter("Game Over");
      display->commit();
      while(1); // note - reboots!
    }
  }
  delay(10);
}
```

In the code, we use rocks to mean falling faces. We start with **INIT_ROCKS** falling, and every **ROCK_INCREASE_SPEED** cycle, we introduce another until **MAX_ROCKS** is reached. Each rock falls from a random position on the top until it reaches the bottom, then it regenerates at a new random position at the top.

The game continues until the collision detection notices that your face collides with a falling face. This is easy to detect because the icons are round. Once a collision happens, Game Over is displayed on the screen and the game restarts. It will need a bit of work to be a fun thing to play – the graphics need an overhaul, a scoring system would be good, and some variation in the gameplay (perhaps a weapon to shoot the rocks). However, this is a tutorial on how to program a game, not a complete game ready to go, so how you progress this game is up to you. 

RESTARTING

When our final game ends, we run:

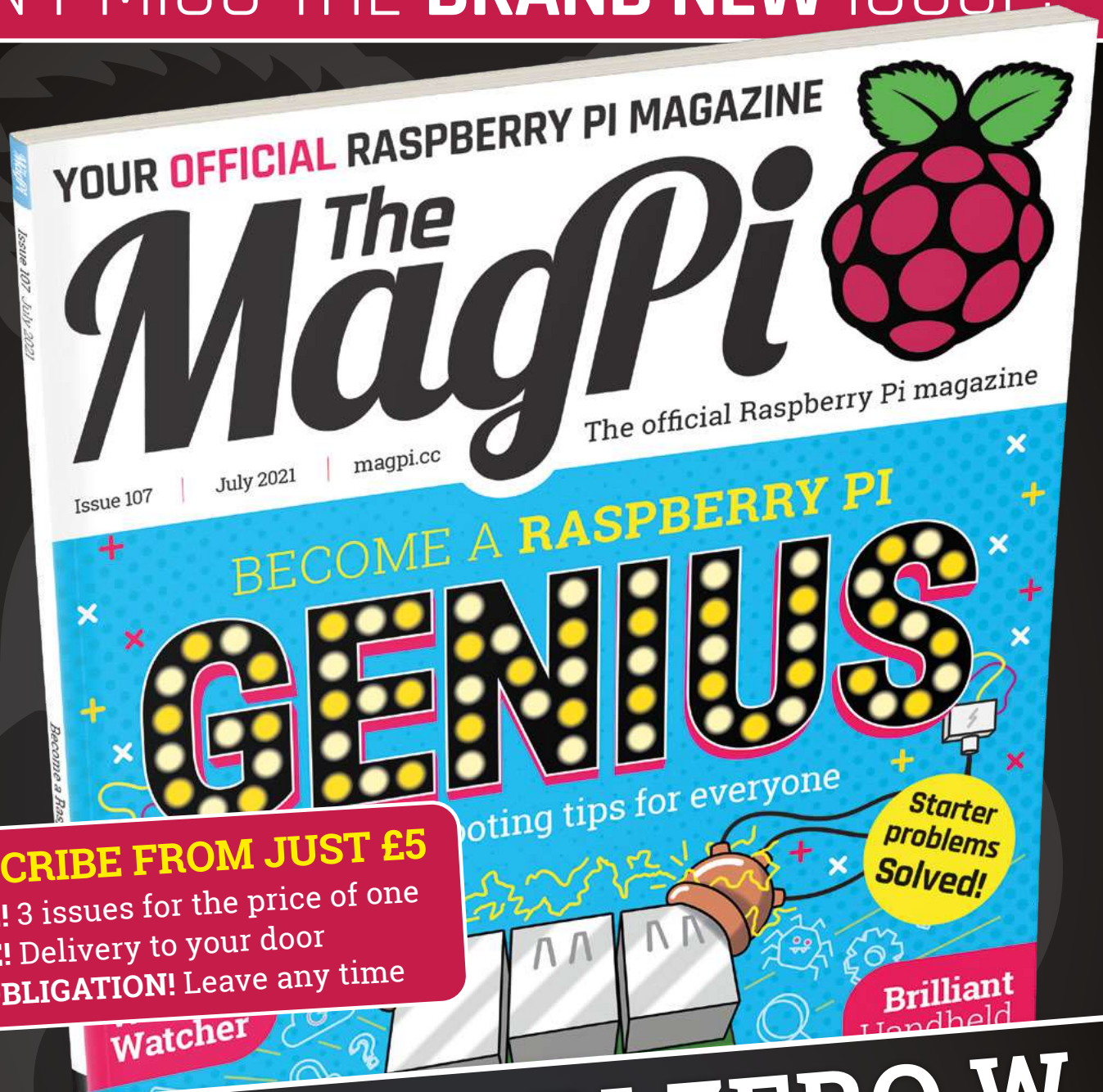
```
while(1);
```

You might think that this creates an infinite loop that just keeps on spinning and doesn't do anything else until the board is restarted. However, that's not what happens. It pauses for a while, then restarts.

The ESP8266 has a watchdog timer that makes sure that something is running. The idea of this is to detect crashes and then restart. In most cases, this is a good idea. It's common for the ESP8266 to manage a WiFi-connected sensor. There's quite a lot going on in the WiFi stack and if anything should go wrong, restarting everything is exactly what we want to happen. However, if for some reason, you want to stop your machine and have it stay stopped, you can do it with:

```
while (1){ESP.wdtFeed();};
```

DON'T MISS THE **BRAND NEW** ISSUE!



SUBSCRIBE FROM JUST £5

- **FREE!** 3 issues for the price of one
- **FREE!** Delivery to your door
- **NO OBLIGATION!** Leave any time

**FREE PI ZERO W
STARTER KIT***

With your 12-month subscription to the print magazine

magpi.cc/12months

* While stocks last



**WORTH
£20**

Buy online: store.rpipress.cc

HackSpace
TECHNOLOGY IN YOUR HANDS

FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG
108



PICO VGA DEMO BASE

Add graphics and sound
to your Raspberry Pi Pico

PG
110



NIBBLE

Learn to code with this
hand-held gaming device

PG
112

EASYEDA

We try designing PCBs
in a web browser

PG
102



BEST OF BREED

Our favourite walking robot
kits for every budget

ONLY THE
BEST

Let's get ready to Robot

Robotics kits for every budget

By Marc de Vinck

 @devinck

Depending on who you ask, the idea of what a robot looks like can be very varied. Kids may think of Gundam-style mech warriors and transformers, while an older generation might think of Rosie the robot from *The Jetsons* or Will Robinson's protector Robot Model B-9. In any event, as you get a little closer to the reality of what robots actually look like, other than those found on a factory floor, they will usually have wheels or tracks for locomotion.

Yes, of course we all know that famous yellow dog-like bot out there, and a few other amazing examples of social-media-loving walking bots. But practical, more everyday-use bots always seem to favour wheels over legs. Mostly because it's definitely easier and cheaper to roll than walk!

Today, for all you readers out there who like to take a different path, we'll be looking at some bot kits that feature legs for locomotion. No wheels here! There are a lot of walking bots out there, and a wide range of prices. We're going to look at a variety of configurations, ranging from two- to eight-legged, and ranging in price from just a few dollars to just about \$500. You can spend a lot more money on more complex walking bots, but we'll save the more expensive varieties for a different Best of Breed.



SmallKat vs Lynxmotion Phoenix

COMMONWEALTH ROBOTICS ♦ \$500 | tindie.com

LYNXMOTION ♦ £216 | robotshop.com

Not all bots that walk need to look like menacing arachnids from outer space. Don't get me wrong: I like that aesthetic, but something a little less intimidating is a nice change.

Introducing the SmallKat, designed by Commonwealth Robotics. This cat-inspired quadruped kit includes everything you need to have your own robotic kitty walking around your studio.

Yes, it's a bit expensive, but not out of the norm for a 16DOF (degrees of freedom) quadruped robot. In that case, it's comparatively inexpensive. The degrees of freedom relate to the robot's movable joints, in this case representing 16 servos for locomotion. Having 16 servos in any robot is going to raise the price substantially. But it also raises the ability for the bot to move more easily and precisely.

The SmallKat is a complete and open-source kit that includes everything, even the 3D-printed cat body. And since it's open-source, you can also download, modify, and print them any way you'd like.



The brain of the robot is a simple ESP32 coupled with an external IMU (inertial measurement unit) for balance. Be sure to check out the link for not only the complete build documentation, but also a lesson plan and educational materials, recorded lectures, and more about building your own robotic cat.

Left ■
Cute kitty or overthrower of humankind?

Below ■
Combine with cobwebs for a Halloween prop

So, you want something a little more intimidating compared to a robo-kitty? Well then, the Lynxmotion Phoenix is a perfect choice. This hexapod frame makes for a robust base for your robot's brain board, sensors, power, and other electronics. All the main components in this base kit are water jet-cut aluminium, making for a solid build.

One major thing to note is this is just the base for your robot. You will need to add just about every other component for it to have any locomotion. That includes purchasing 16 Hitec servos, which can add up quickly. Having so many servos in one robot can be difficult to manage, but Lynxmotion does provide the

software to make it much easier to do some basic locomotion and path planning. If you want to build a hexapod, this is a good choice for beginning with a solid foundation. →



VERDICT

SmallKat

Not all walking bots have to be spider-like.

9/10

Lynxmotion Phoenix

Very creepy and very well-built.

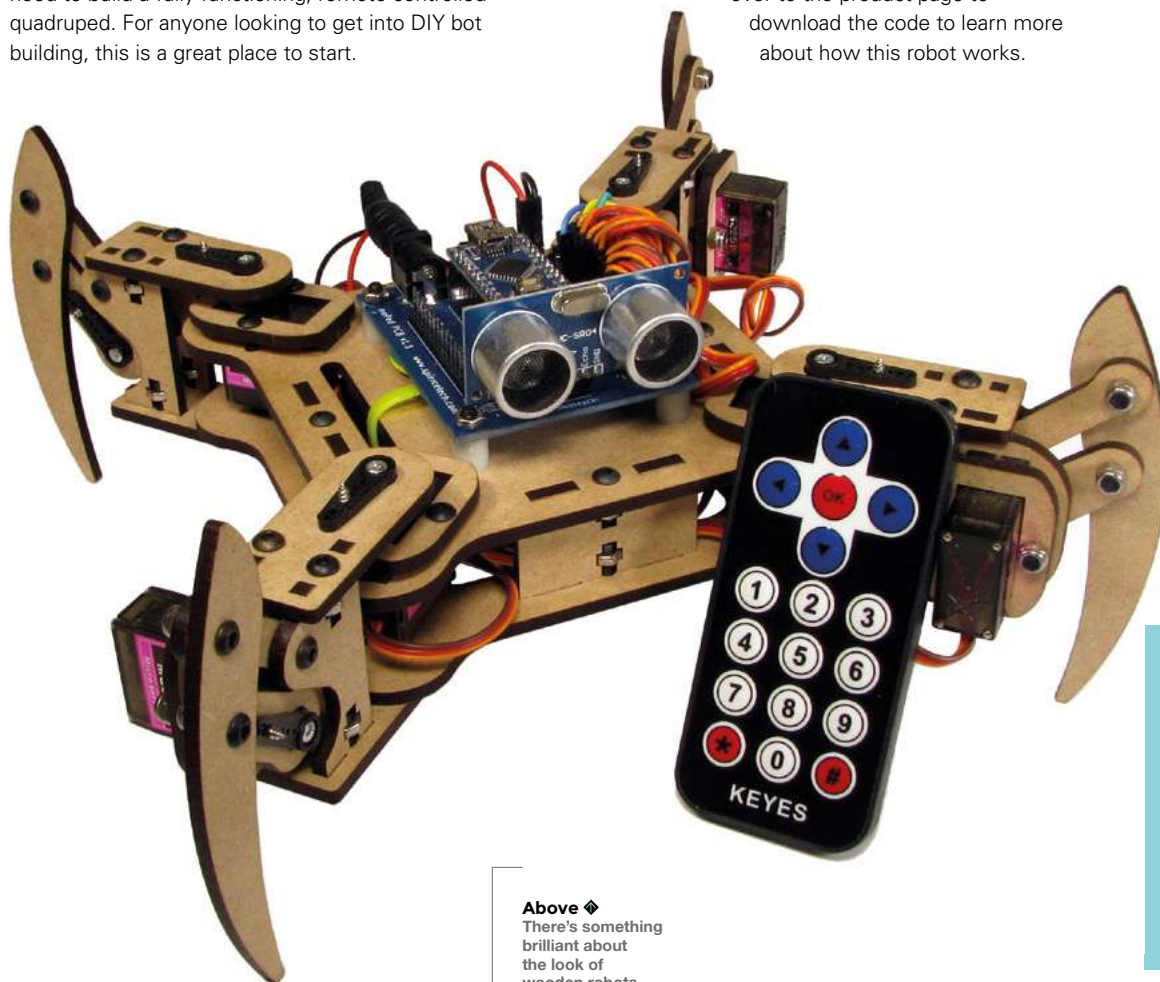
10/10

mePed v2 quadruped walking robot

MEPED ♦ \$79.95 | tindie.com

The mePed v2 quadruped walking robot is one of our favourite bots in this roundup. Not because it's the fastest or strongest, but because of the completeness of the kit and especially the price. The kit includes everything you need to build a fully functioning, remote-controlled quadruped. For anyone looking to get into DIY bot building, this is a great place to start.

The mePed v2 is also open-source, so you can easily modify the structure, electronics, and code to your specifications. You can do a lot of things with just the on-board Arduino-compatible microcontroller and ultrasonic sensor, but you could also upgrade it to something more powerful, like the new Pico. Head over to the product page to download the code to learn more about how this robot works.



Above ♦
There's something brilliant about the look of wooden robots

VERDICT

mePed v2 quadruped walking robot

Not many walking bots include everything, but this one does!

10/10

DIY B/O Spider Robot

DFROBOT ♦ \$5.90 | dfrobot.com

Yes, I know, this looks like a toy. And in fact, it is just that, a toy. But for such a low price, you'd be surprised at how much the DIY B/O Spider Robot can do with a few additional parts. Right out of the box, and after some simple assembly, the little spider-bot can run forwards and backwards, but that's about it. It's a neat little desktop toy and educational kit.

But what we really like about the kit are all the accessories and modifications that are out there on the internet. People have added microcontrollers, like the recommended Romeo controller. And others have gone even further by adding extra sensors, WiFi, BTLE, and even apps for controlling this robot



with your smartphone. The DIY B/O Spider Robot is a fun and incredibly affordable little spider platform. You'd be hard-pressed to find a less expensive eight-legged robot.

Left ♦
Perhaps the cheapest way to build a walking robot

VERDICT

DIY B/O Spider Robot

Interested in walking bots? Get this!

9/10

SIX – Hexapod Robot Kit

EZ-ROBOT ♦ \$449.99 | ez-robot.com

The aptly named **EZ-Robot Revolution SIX WiFi Hexapod** is a very user-friendly robotics kit. At its core are the EZ-Bits, which allow you to customise and build the bot as needed. The joints are all powered by twelve heavy-duty servos, and the manufacturer provides an easy-to-use program, EZ-Builder Robot, to control them all. You can also add more sensors to the bot through the EZ-Clip system which integrates them into the main body.

The software provides basic manoeuvrability and the ability to run animations, or scripted poses, to give your bot a little life. You can also use your smartphone to control the bot, and it can even stream live video from the on-board camera, or use it to do some basic image detection and object tracking. The robot also features a built-in speaker, can be programmed in multiple

languages, and even has speech recognition. The EZ-Robot site provides lots more information, along with example use cases. Be sure to visit it for a lot more information about this customisable six-legged bot. →



Left ▣
Build your own hexapod with this user-friendly kit

VERDICT

SIX – Hexapod Robot Kit

An easy-to-build bot.

9/10

Lynxmotion Biped Robot Scout

LYNXMOTION ♦ £96 | robotshop.com

When most people think of a walking robot, they think of a bipedal design. But most 'walking' robots have more than two legs. That's because bipedal locomotion is really hard! The Lynxmotion Biped Robot Scout solves the complicated walking mechanics in a nice 6DOF form factor for each leg.

Like most other servo-driven robots, the Scout comprises just the frame and servo mounts. It does not include all the required servos and associated electronics, and those costs can add up fast. But once you have gathered enough servos, twelve for the Scout, and picked an appropriate microcontroller, you'll be well on your way to getting your robot up and walking. □



VERDICT

Lynxmotion
Biped Robot
Scout

Bipeds can be more economical when it comes to walking bots.

8/10

Above □
Walking on two legs can be hard, but gives your robot a more human look

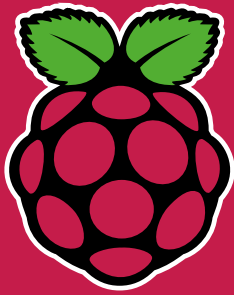
LIGHT CHASER BEAM ROBOT KIT

DFROBOT ♦ \$14.90 | dfrobot.com

Who said the legs of a walking robot had to articulate? Not me! The Light Chaser Beam Robot Kit is a more advanced brushbot which adds some sophistication with a BEAM circuit. Once turned on, this little bot will scurry around looking for the brightest source of light. These are fun to build and a great way to get kids introduced to electronics.



THE *Official*

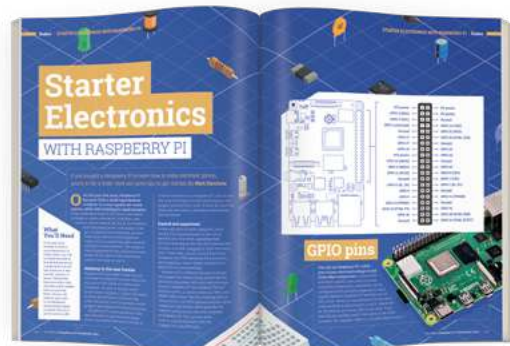
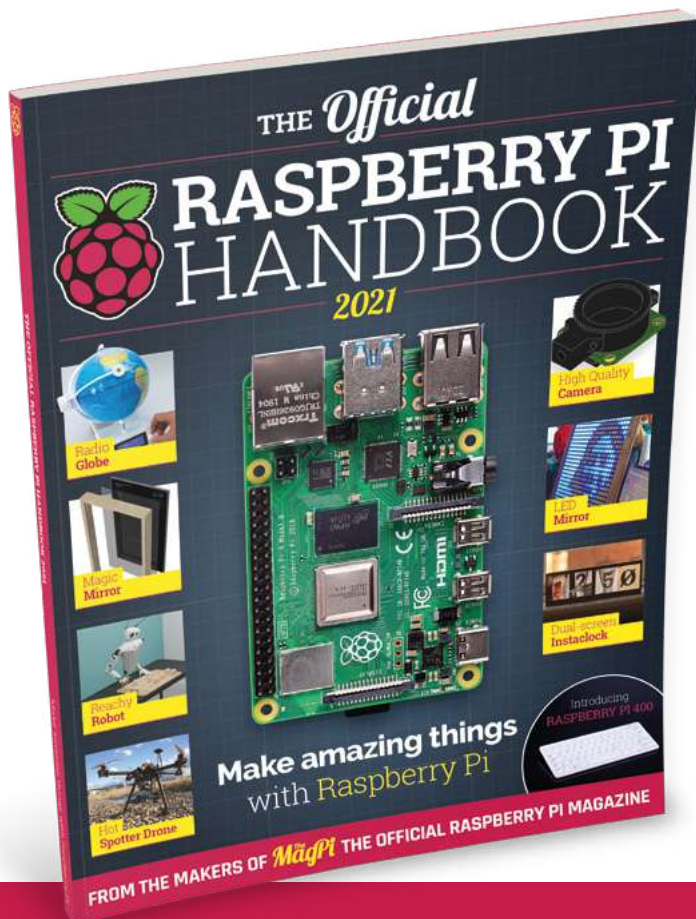


RASPBERRY PI HANDBOOK

2021

200 PAGES OF RASPBERRY PI

- Get started with Raspberry Pi, electronics, and more
- Be inspired by incredible projects made by other people
- Learn how to code and make with our step-by-step tutorials
- Find out about the top kits and accessories for your projects



Buy online: magpi.cc/store

Pico VGA Demo Base

Sound and video for your microcontroller

PIMORONI ♦ £18 | pimoroni.com

By Ben Everard

🐦 @ben_everard

The Raspberry Pi Pico has a software development kit (SDK) packed full of useful features to help you create your programs. There is also a Pico Extras repository (hsmag.cc/pico-extras) that has a bunch of features that haven't yet made it into the final SDK. In here, you'll find things such as a library for VGA graphics, an audio system, and an SD card library. There's also a GitHub repository with some example applications that use these features at: hsmag.cc/PicoPlayground. These examples range from simple things like a sine-wave generator to more complex things like a movie player. Pico Demo gives you the ability to experiment with these.

Pico Demo is based on the reference design from the *Hardware design with RP2040* book (available at

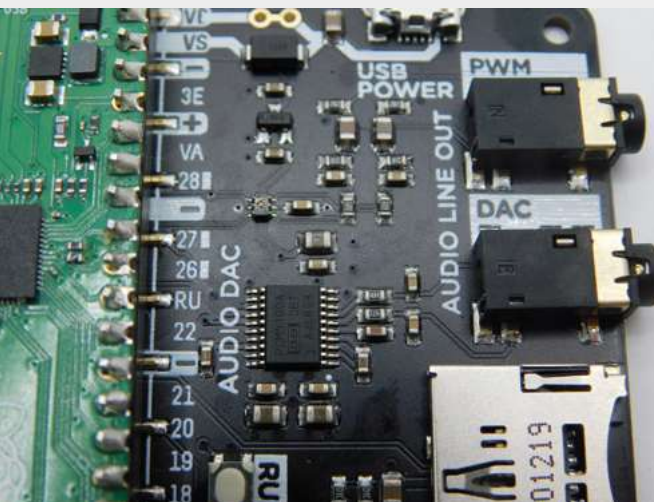
hsmag.cc/RP2040_HardwareDesign), so you can delve into the design decisions in the hardware if you want to.

This author has set up almost the entire hardware of this board on a breadboard previously, so he can attest that it's so much easier and nicer to have it all on a PCB ready to go. The VGA section, particularly, is electrically noisy and fiddly to set up and run without a dedicated board.

The main limitation with this board is that it's so packed full of outputs that there's very little space left for other inputs or outputs. In fact, with everything running, the only option for inputs are the three buttons (which themselves are on lines shared with the VGA output). You do have the option to use two pins from the SD card header, but only if you don't use the SD card.



Right ♦
The board
is about the
same size as
a credit card



The only other available input/output option is the USB port. Since Pico has support for USB host mode, you can use this with USB peripherals. The Pico VGA Demo Base has a separate USB port for power; it's easy to add a micro USB-to-USB converter to let you plug in keyboards, mice, or gamepads.

The original hardware was designed to show off Pico's processing capabilities in the style of the Demoscene. For that, it's a great option. It will also make a great base for a games machine – just add a USB controller and VGA monitor.

While only two GPIO pins are available (if you don't use the SD card), they do happen to function as an I2C bus, so if you do need to add more features, it could be through this. You could even pop an I2C port expander on it if you need more GPIOs. A slight annoyance – if this is the route you want to go down – is that the 3V line isn't broken out, so you'll have to either add another regulator to the 5V (VBUS) line, or make an extra connection into the 3V pin.

PICO EXTRAS

The Pico Extras GitHub repository is for Pico SDK features that aren't yet finalised. They should work, but haven't had as much love as the features in the main SDK. Over time, they should stabilise and move to the main SDK.

It's possible that when you read this, the video, audio, and SD card features will have migrated from Extras into the main SDK, so take a look there if you can't find them. Of course, at this point in the future, there might be even more exciting stuff in Pico Extras, so it's worth checking out even if you now have what you need in the main SDK.

The analogue signals for the graphics are created by using multiple GPIO lines for each colour, and putting them through a resistor ladder so that one signal has twice the influence of the next, and that has twice the influence of the one after that, and so on. There are five GPIOs for red, five GPIOs for green, and five GPIOs for blue. This means you can display 15-bit colour. You could remove some of these GPIOs to give space to connect more hardware at the expense of colour depth; however, this would entail removing some of the surface mount resistors on the board.

While this hack to enable more GPIO connections is possible, it's not really the point of the board, which is for projects that want to show off outputs. This is for people who want to squeeze all the available graphics power out of a microcontroller and show it off to their friends.

Plugging a USB games controller and VGA monitor into a microcontroller shouldn't be a thing you're contemplating doing. If you want to interact with those devices, it's almost certainly a better idea to use a more powerful single-board computer (such as a Raspberry Pi). However, it is possible with Pico, and that's a good enough reason for us.

In short – the Pico VGA Base is a board for people who want to make Pico do things a microcontroller shouldn't really do, just for the sake of having done it.

As microcontrollers go, Pico does have impressive capabilities for video and sound, and Pico VGA Demo Base makes it easy to show these off – just add software. We can't wait to see what you create. □

Left ♦

The Pico Demo VGA Base has a choice of PWM or I2S audio output

Below ♦

A resistor ladder is used as a digital-to-analogue converter to send the signal to the VGA connector

VERDICT

Show off Pico's sound and graphics abilities

9/10



CircuitMess Nibble

Build your own portable gaming machine

CIRCUITMESS ♦ £47.99 | circuitmess.com

By Ben Everard

🐦 @ben_everard

Nibble is a games console for makers, or anyone else who likes to understand more about how technology works. While most games machines come as pre-packaged devices with strict instructions not to open them, Nibble comes as a collection of parts for the user to assemble.

The main PCB comes partially assembled, with the surface-mount parts already in place. The recipient, then, has to solder on the through-hole parts. They're all quite chunky and we wouldn't expect this to be a problem even for beginner solderers – in fact, Nibble could be a great introduction to soldering for someone looking to learn how to assemble electronics. Once the through-hole parts are in, there's a laser-cut enclosure to bolt together. Pop in a few batteries and you're ready to play games.

Below ♦
The acrylic enclosure feels sturdy enough to carry around



The real purpose of this, though, isn't about putting it together: it's about using it as a platform for creating your own games.

You can program Nibble either with the visual CircuitBlocks editor or the Arduino IDE. Peek behind the curtain and they're both the same environment really, as CircuitBlocks creates C++ code that can then be compiled by the Arduino toolchain.

Unfortunately, the documentation for Nibble's Arduino library (based on the CircuitOS library: hsmag.cc/CircuitMess) is fairly thin on the ground. Even if you don't plan on using the CircuitBlocks graphical editor, it can be worth having a bit of a click through to see the code it generates, which shows how to interact with the Arduino library. We've also created a 'getting started' tutorial (see page 98).

There are quite a few portable games consoles aimed at makers now. Broadly speaking, you can split them into two distinct categories: old-school, which are typically 8-bit with monochrome screens, and new-school, which are 32-bit with colour screens. Nibble fits firmly in the new-school category. It's got a fast ESP8266 32-bit processor with 80kB of RAM and 4MB of storage, and it's got a 128×128 colour screen. This all gives you enough power to really make use of the colours and screen without having to worry about being thrifty with clock cycles or memory.

The examples that come preloaded on Nibble give a good idea of what's possible. Space Invaders is a particular favourite here in HackSpace mag towers, but there's also Pong and Asteroids. There's a space on the CircuitMess website for community creations. However, as yet, there's nothing for Nibble. Delve a little deeper into the website and you'll find the forum where (among the usual questions) a couple of users have uploaded their creations: a Sokoban clone and Anarch (a DOOM-like FPS).

There's not the huge range of games ready-to-go, as there is on CircuitMess's previous console (the MAKERbuino), but this is enough to keep you entertained while you build your own games. Perhaps more will appear in the future.

**// You get quite a bit of
— hardware for your money
and there's quite a bit to
play with //**

As it currently stands, if you flash the default firmware onto Nibble, then you get a selection of games. However, if you load additional games onto it, you will only have that one game available – it's not possible to add new games to the launcher. There's no off-chip storage (such as an SD card), so everything has to fit into the 4MB storage on the ESP8266. That means that even if software support does come out to allow loading multiple games, you're not going to be able to fit a lot on. It's pretty easy to flash new games to the device, however, so it's not too big an inconvenience.

As Nibble is based on the ESP8266 SoC, it does have WiFi. You could use this for collaborative games, online high scores, or a whole bunch of other possibilities, but there aren't any examples of this at the moment.

The programming side of Nibble is let down a little by the lack of documentation. There's no clear source of information on what particular functions do, or what's available other than digging through the library's header files. We didn't find that it took too long to get to grips with the system, but less



experienced programmers might find it a little confusing. It isn't too big an issue because there is the CircuitBlocks environment that has everything available as pre-created blocks that you can drag and drop around.

This is all about documentation and software. The hardware was easy to assemble and feels solid in our hands. It's something that we'd feel happy putting in our pocket or chucking in our bag and being confident that it'd still be working when we took it out at the end of the day.

We've had fun testing out Nibble. You get quite a bit of hardware for your money and there's quite a lot to play with – probably more in software than in hardware. If you're looking for a way to build your own on-the-go games, then this is a good option. □

Above ♦
Three AA batteries
power your on-the-
go gaming

VERDICT
Solder your
own portable
games
machine.

9/10

EasyEDA Standard

Create a PCB from your web browser

LCSC  Free | easyeda.com

By Ben Everard

 @ben_everard

EasyEDA is an electronic design automation (EDA) tool for designing circuits and creating PCBs. There is also a simulation mode, but we won't be looking at that here as it's a subject in itself.


You can use the online version of the app at **easyeda.com** without installing anything, or you can download an offline version. We used both, and they both worked in much the same way. For testing things out, the online version is a great way to get started, but you might find it more convenient to use the offline version and not have to rely on an internet connection.

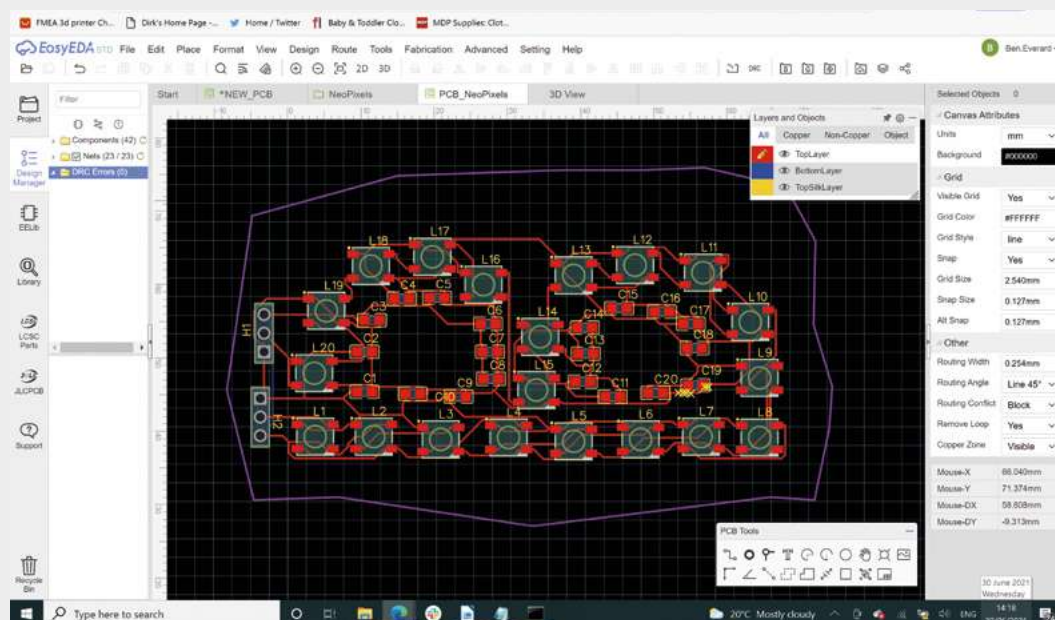
The basic workflow will be familiar to anyone who has worked with PCBs. First, you design your schematic, then you implement this circuit in a PCB.

Creating a schematic is basically just selecting the components you want and linking them together.

A big part of this is what schematic symbols already exist and what you have to create yourself. For example, if you wanted to add an RP2040 microcontroller to your project, can you just select the item and place it, or do you have to first create a symbol, label the pins, and potentially create a PCB footprint? EasyEDA comes with a preset library of a few common components in multiple footprints. It's also linked to a vast online library of parts, including those by LCSC Electronics, as well as user-contributed libraries. You can import your symbols from other popular EDA tools.

At this point, we should stop and say that EasyEDA is linked to LCSC Electronics (a component supplier)

Right  We placed our components where we wanted them and then let the auto router do the hard work



and JLCPCB (a PCB fabricator). Whether or not you consider this a good thing or not depends on your view of bundled products. It means that you can use the LCSC catalogue to select components, and it will create a shopping cart for you. You can also export your PCB designs straight to the JLCPCB store and get them made. However, despite this link, it doesn't lock you in. You can use any components whether or not LCSC Electronics supply them, and you can export the final design files in a format (Gerber) that you can take to any PCB design fabricator.

Select parts from the LCSC Electronics catalogue and you can also export everything to have your boards fabricated and assembled for you. Your design will turn up at your doorstep as a ready-to-go device.

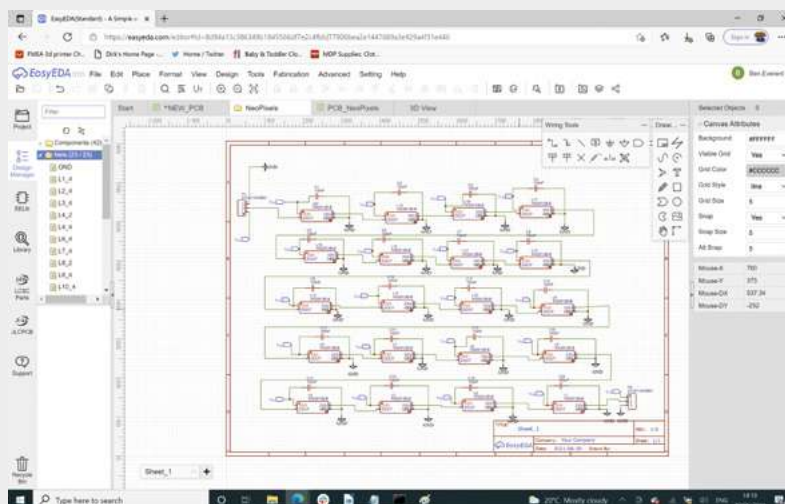
Creating the schematic was straightforward. Pop the components where you want them, and draw on the links you want. The only frustration we had designing a schematic was when moving a symbol that had a connected trace – the trace would move with the symbol. However, if it moved past another pad, it sometimes inadvertently picked up another connection to this pad. This was hard to spot and could have left errant traces in our final design.

Once you've got your schematic created, it's time to create a PCB. Parts are linked to PCB footprints when you place them in the schematic, so there's no intermediate step between the two.

We tested this out by creating a PCB containing an 'artful' arrangement of WS2812B LEDs and associated capacitors in the shape of a letter B. Clicking on 'Create PCB from schematic'

pre-populates a PCB with your components. You can place them where you want them, and then you need to create PCB traces that act like wires linking everything together. You can do this either manually or by using the auto router. The auto router can work in the cloud, and we had some success with this, but it looks like there's a limit on the cloud resources available for this – we sometimes got a message telling us to try again later. Fortunately, you can get around this by running the auto router locally. Even when using the web-based software, you can download and run the router and it will all connect up.

While our example project wasn't the most complex, the auto router handled it all without any problems. We just popped the components where we wanted them, pressed the button, and out popped a routed PCB.



The PCB design tool isn't brilliant at creating custom or unusual shapes. However, you can import an SVG, which means you can use Inkscape or your favourite vector image editor to create your board outline (or other layers) and import these into EasyEDA.

EDA tools are used by many people, from those creating simple breakouts for surface-mount components, to those creating hugely complex

products. Here at HackSpace magazine, we're far closer to the former than the latter. We're not going to attempt to make a product.

For our simple use, we were able to pick up EasyEDA and quickly

work through to a finished and fabricated PCB. It's far quicker and more intuitive than other options we've tried. By EDA standards, it's unimposing, and most of the useful tools and things we needed were easy to find. There's a video series that might be useful if you're new to PCB design: hsmag.cc/EasyEDA.

The world of PCB design can be a bit opaque to outsiders. Having an online tool that's reasonably easy to use and linked directly to a fabricator (no wondering what Gerber options to select, or missing files from the zip file) makes getting your first few PCBs completed and fabricated much easier. While this is integrated with JLCPCB, it isn't tied to them, which makes us much more confident about using it. This is definitely a PCB designer we'll be exploring in more depth in future issues. □

Above ♦
The circuit is just WS2812B LEDs and associated capacitors

Creating the schematic was straightforward. Pop the components where you want them, and draw on the links

VERDICT

Create and fabricate PCBs without leaving your web browser.

9/10

issue
#46

ON SALE
19 AUGUST

CITIZEN SCIENCE

ALSO

- 3D PRINTING
- RASPBERRY PI
- ARDUINO
- T-SHIRTS
- AND MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe



A person is shown from the chest up, wearing a costume with large, ornate, light-colored shoulder pieces that have a scalloped, flame-like design. They are wearing a blindfold made of several horizontal strips of light-colored material. Their hair is dark and long. The background is dark and out of focus.

"The designs in games are absolutely perfect for cosplayers. They're so impressive; there are so many different elements that you can use to show off. It makes for some really great costumes."

WILLOW CREATIVE

10 MILLION+ PRODUCTS ONLINE | 1,300+ INDUSTRY-LEADING SUPPLIERS | 100% FRANCHISED DISTRIBUTOR

From Maker to Manufacturing

**FREE
SHIPPING**

ON ORDERS OVER
£33 OR \$50 USD*



0800 587 0991
DIGIKEY.CO.UK



*A shipping charge of £12.00 will be billed on all orders of less than £33.00. A shipping charge of \$18.00 USD will be billed on all orders of less than \$50.00 USD. All orders are shipped via UPS, Federal Express, or DHL for delivery within 1-3 days (dependent on final destination). No handling fees. All prices are in British pound sterling or United States dollar. Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2021 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA

ECIA MEMBER
Supporting The Authorized Channel